A REPORT

ON

ANALYSIS OF PHYSIOLOGICAL SIGNALS FOR THE STATE MONITORING OF SUBJECTS

BY

Name	**ID**
Srujan Deolasee	2019A7PS1139P
Akshat Jain	2019B1A30987P
Aditya Shrivastava	2019AAPS0334H
Geet Gautam	2019A3PS0261H

Prepared in fulfillment of the
Practice School-I Course Nos.
BITS C221/BITS C231/BITS C241

AT

CSIR-CEERI Chennai

A Practice School-I Station of

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

July, 2021

A REPORT

ON

ANALYSIS OF PHYSIOLOGICAL SIGNALS FOR THE STATE MONITORING OF SUBJECTS

BY

Name	**ID**	**Discipline**
Srujan Deolasee	2019A7PS1139P	Computer Science
Akshat Jain	2019B1A30987P	Electrical and Electronics + Biological Sciences
Aditya Shrivastava	2019AAPS0334H	Electronics and Communication
Geet Gautam	2019A3PS0261H	Electrical and Electronics

Prepared in fulfillment of the
Practice School-I Course Nos.
BITS C221/BITS C231/BITS C241

AT

CSIR-CEERI Chennai

A Practice School-I Station of

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

July, 2021

Acknowledgements

The completion of this report would not have been possible without the assistance of many people whose names may not have been mentioned. Their contributions are appreciated and acknowledged. We would first and foremost like to thank the organization, Council of Scientific & Industrial Research-Central Electronics Engineering Research Institute (CSIR-CEERI) and all its members for facilitating our work here. Our immense gratitude to Mr Palaniandi of CSIR-CEERI, Chennai, for hosting us at CEERI, Chennai and providing the necessary facilities. We would also especially like to thank our mentor, Dr Madan Kumar Lakshman, Senior Scientist, CSIR-CEERI, Chennai, for giving us this project, guiding us and giving us access to the proper resources. We are much obliged to the Practice School Division of Birla Institute of Technology and Science, Pilani (BITS Pilani) for allowing us to intern at CSIR-CEERI, Chennai. We want to thank Dr. Amalin A Prince, Professor-in-Charge, Practice School-1 CSIR-CEERI, Chennai, for his constant guidance and supervision.

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)

Practice School Division

Station: CSIR - CEERI **Center:** Chennai
Duration: 53 Days **Date of start:** 31 May 2021
Date of Submission: 22 July 2021
Title of Project : ANALYSIS OF PHYSIOLOGICAL SIGNALS FOR THE STATE MONITORING OF SUBJECTS

Name	ID	Discipline
Srujan Deolasee	2019A7PS1139P	Computer Science
Akshat Jain	2019B1A30987P	Electrical and Electronics + Biological Sciences
Aditya Shrivastava	2019AAPS0334H	Electronics and Communication
Geet Gautam	2019A3PS0261H	Electrical and Electronics

Mentor: Dr. Madan Kumar Lakshmanan, Principal Scientist, CSIR-CEERI, Chennai
PS Faculty: Dr. Amalin A Prince

Keywords: PPG, Fatigue, KNN, SVM, RF, Naive Bayes
Project Area : Machine Learning Classification

Abstract: In this report, we present our examination and study of Person State Classification. We first provide a review of the PPG signal used for collecting data, explaining the process of feature extraction in brief. The extracted features are cvHR, HF Power, LF Power, LH Ratio, mHR, RMSSD, pNN20, pNN30, pNN40, pNN50, stdHR, and T Power. We use these features for training traditional machine learning algorithms like kNN, Naive Bayes, RF, SVM for state classification. From our results, we conclude that SVM and RF perform better than kNN and Naive Bayes, achieving over 90% accuracy.

Aditya, Akshat, Geet, Srujan
Signature of Students Signature of PS Faculty

Table of Contents

Section 1: Introduction

According to a study conducted by Central Road Research Institute (CRRI), around 40% of highway accidents in India occur due to drivers dozing off. Fatigue and drowsiness are some of the most common reasons behind road and industrial accidents. Hence, there is a dire need for state monitoring and classification systems with high accuracy rate. However, they should also be cost-friendly and feasible to use in the specific environment of the user. They should also be easy to wear and remove without assistance.

The project uses data collected from a Samsung Gear S3 band, a non-invasive sensor, to detect the state of the person. Our work primarily was about building machine learning models that will be used to classify the state of the person (normal/alert vs drowsy/fatigued) using the data that was collected by the sensors. While many researchers used three types of physiological signals simultaneously, namely Electroencephalography (EEG) signals, Electrocardiography (ECG) signals, and Photoplethysmography (PPG) signals to measure drowsiness levels using machine learning models, we have only used PPG signals to achieve a good accuracy of state prediction. The PPG signals are captured from different subjects under states such as sedentary, sleep and mental fatigue. Vital heart rate and heart rate variability measures are derived from the PPG signals. This data was then used to train machine learning algorithms after which information on the person's state was predicted.

SVM, KNN, RF and Naive Bayes classifiers were used to classify the preprocessed data. Python and its libraries were used to visualize, plot, preprocess and train and test Machine Learning algorithms.

Section 2: Physiological Signals

Fatigue is one of the significant causes of road traffic accidents. One way to monitor a driver's drowsiness is by monitoring various signals like EEG/ECG/PPG.

EEG is a non-invasive electrophysiological monitoring method that represents the macroscopic activity of the surface layer of the brain with the help of electrodes. Electrocardiogram (ECG) is a graph of voltage versus time of the heart's electrical activity with the help of non-invasive electrodes. The electrical changes detected by the electrodes result from depolarization followed by the repolarization of the cardiac muscle during one heartbeat. PPG is based on the optical detection of blood volume changes in the microvascular bed of tissue. Although EEG is a very accurate indicator of driver fatigue, existing EEG systems are impractical in actual driving situations. In addition, these devices have very limited wearability as can be seen evidently in Figure 2.1.

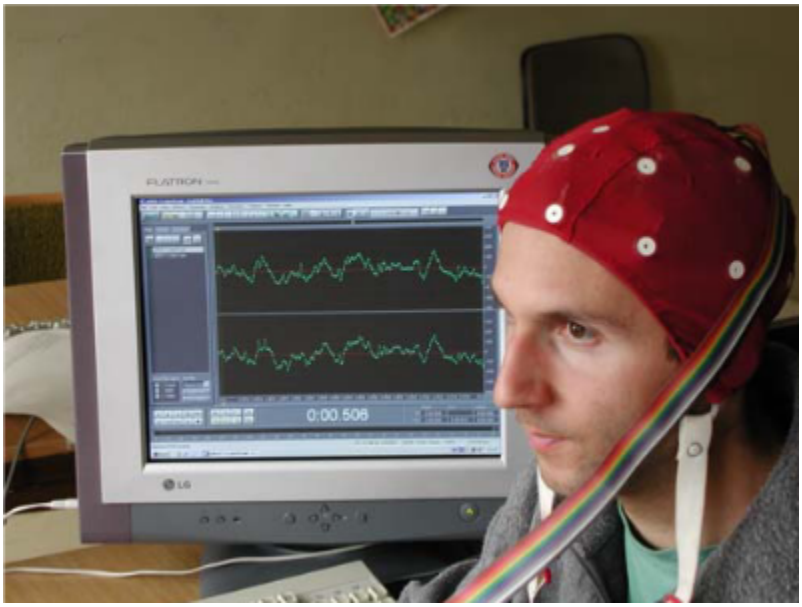


Figure 2.1: EEG Recording Setup[1]

During stress, fatigue, drowsiness, there are alterations in the Autonomic Nervous System (ANS). State of wakefulness is characterized by increased sympathetic activity or a decrease of parasympathetic activity. In contrast, extreme relaxation states are characterized by increased parasympathetic activity or decreased sympathetic activity. ANS activity can be measured with the help of HRV signals obtained using surface ECG. It is known that power on the low frequency (LF) band (0.04-0.15Hz) is mainly considered a measure of sympathetic activity. In contrast, power on the high frequency (HF) band (0.15-0.4 Hz) is considered of parasympathetic origin in classical HRV

analysis [2]. The LF/HF ratio measures balance between sympathetic and parasympathetic systems.

One of the studies proposed a method that integrates features of electrocardiography (ECG) and electroencephalography (EEG) to improve detection performance. Some of the EEG features extracted included time domain statistical descriptors, complexity measures and power spectral measures. Features extracted from the ECG signal included heart rate (HR) and heart rate variability (HRV), including low frequency (LF), high frequency (HF) and LF/HF ratio. Significant features of both modalities (EEG and ECG) are then combined to investigate the improvement in performance using support vector machine (SVM) classifiers[3].

The channel reduction analysis in the study revealed an acceptable level of accuracy (80%) that could be achieved by combining just two electrodes (one EEG and one ECG), indicating the feasibility of a system with improved wearability compared with existing EEG systems.

For our project, we worked only with the PPG signal data due to the ease of its collection.

2.1 PPG Signals

A photoplethysmogram (PPG) is an optically obtained plethysmogram that can be used to detect blood volume changes in the microvascular bed of tissue. A PPG is often obtained by using a pulse oximeter which illuminates the skin and measures changes in light absorption[4]. A conventional pulse oximeter monitors the perfusion of blood to the dermis and subcutaneous tissue of the skin. Photoplethysmography measures the volumetric change of the heart by measuring light transmission or reflection. As the heart contracts, blood pressure within the left ventricle – the main pumping chamber – increases. This increase forces a pressurized "pulse" of blood into the body's arteries, which causes them to swell slightly before returning to their previous state. By simply shining a light on a patch of skin with an LED light source, the increased pulse pressure will cause a measurable difference in the amount of light reflected onto or transmitted through to a light sensor. The LED light should be placed in an area where arteries are close to the skin, such as a fingertip or an earlobe. The amplitude of this signal is directly proportional to pulse pressure – the higher the peak, the stronger the pulse. Even though the signal can be less apparent if measured on the skin at a point far from the heart (such as a toe), the change in pressure is still enough to expand these arteries to a measurable degree.

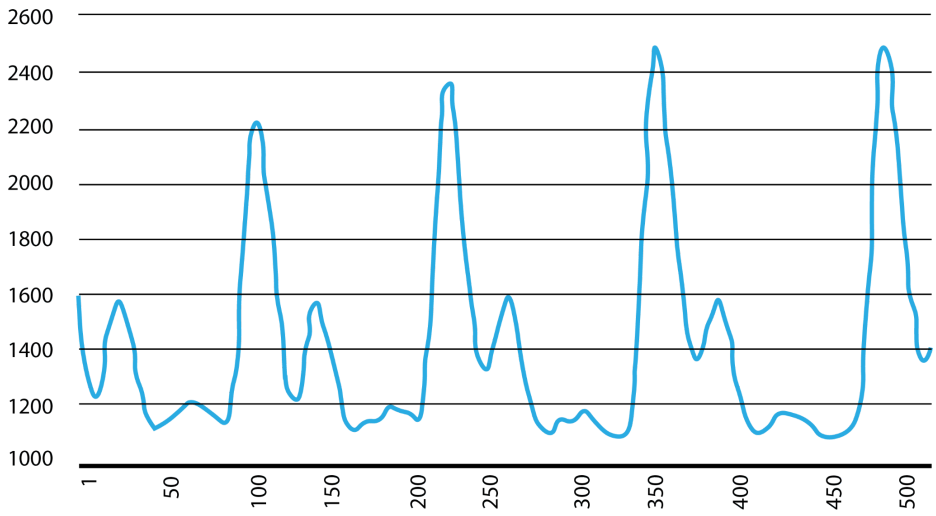


Figure 2.2: Example of typical PPG signal[5]

Each peak in the resulting signal, like shown in Figure 2.2, can be identified by a particular heart rate algorithm that can ultimately determine the amount of time that occurs between each successive peak – giving another measure of heart rate. Because PPG is an indirect technique (that does not record the heart's actual activity occurring at its source), it is not always accurate to the millisecond level, yet is an easily applied way in which to measure an individual's heart rate[5]. PPG technology is becoming more readily available, inexpensive, convenient, and easily integrated into portable devices. Recent advances include the development of smartphones and wearable devices that collect pulse oximeter signals. The PPG data used for this project has been generated using a Samsung Gear S3 band.

2.2 Principle of PPG Signals:

PPG makes use of low-intensity infrared (IR) light. When light travels through biological tissues it is absorbed by bones, skin pigments and both venous and arterial blood. Since light is more strongly absorbed by blood than the surrounding tissues, the changes in blood flow can be detected by PPG sensors as changes in the intensity of light. The voltage signal from PPG is proportional to the quantity of blood flowing through the blood vessels. Even small changes in blood volume can be detected using this method, though it cannot be used to quantify the amount of blood. A PPG signal has several components including volumetric changes in arterial blood which is associated with cardiac activity, variations in venous blood volume which modulates the PPG signal, a DC component showing the tissues' optical property and subtle energy changes in the body. Some major factors affecting the recordings from the PPG are site of measurement

and the contact force between the site and the sensor. Blood flow variations mostly occur in the arteries and not in the veins[6].

2.3 PPG Signal Processing:

The data was collected using the Samsung Gear S3 device. The subject wore the band on his/her hand. The subjects were asked to play a game for 30 minutes, and data was collected under various stages such as sedentary, sleep, and mental fatigue. The n-back test suggested by Tanaka et al[7] was used to simulate mental fatigue. The device had integrated PPG sensors operating in the green channel to capture PPG signals at a sample rate of 20 Hz. For analysing the data for motion artefacts, the pre-processed data was passed through Butterworth lowpass filter, Hampel filter, and Savgol filter. The signal was transformed into the frequency domain using the Welch transform. Signals of smaller time segments (15 sec) were used for this purpose. Each partition is called a time bin. Time bin where a clear peak in the frequency range of 0.7 to 2.2 Hz was tagged as having a good pulse quality index. Only these bins were retained for subsequent analysis. Various time and frequency domain measures were derived from the signals. These measures were then used to train machine learning algorithms to predict the state of the subject.

Section 3: Features

In machine learning, feature means property of the training data. Or one can say a column name in the training dataset. Briefly explained, a feature would be the input one has fed to the system and the label would be the output expected.
The features can be of different types-

1. Simple Supervised learning, where they are simple values like numbers and characters.
2. In unsupervised learning, the model is itself trained to recognise the features and work on it.

In this project, some data is already tagged with the correct answer. After that, the algorithm is further provided with a new set of data, so that the supervised learning algorithm analyses the training data and produces a correct outcome from the labelled data.

3.1 Feature Engineering:

Feature engineering involves leveraging data mining techniques to extract features from raw data along with the use of domain knowledge. Feature engineering is useful to improve the performance of machine learning algorithms and is often considered as applied machine learning. Feature selection, construction, transformation, and extraction are some key aspects of feature engineering.

3.2 Feature Selection

The process of selecting those features which contribute most to the prediction variable or output in which one is interested in. Having irrelevant features in the data can decrease the accuracy of the models and make the model learn based on irrelevant features.
Importance of feature selection -

1. Reduces overfitting : Less redundant data means less chances of machine making decisions based on noise.
2. Improves accuracy : Less noise means increased model accuracy.
3. Reduces training time : Fewer and effective data points reduce algorithm complexity and training time.

Feature Selection Methods -

1. Univariate Selection :

Statistical tests can be used to select those features that have the strongest relationship with the output variable.The scikit-learn library provides the SelectKBest class that can be used with a suite of different statistical tests to select a specific number of features as shown in Figure 3.1.

	Specs	Score
13	ram	931267.519053
11	px_height	17363.569536
0	battery_power	14129.866576
12	px_width	9810.586750
8	mobile_wt	95.972863
6	int_memory	89.839124
15	sc_w	16.480319
16	talk_time	13.236400
4	fc	10.135166
14	sc_h	9.614878

Figure 3.1 Example of univariate selection showing top 10 best features using SelectKBest class[8]

2. Feature Importance :

Feature importance gives a score for each feature of your data, the higher the score more important or relevant is the feature towards the output variable.Feature importance is an inbuilt class that comes with Tree Based Classifiers as shown in Figure 3.2.

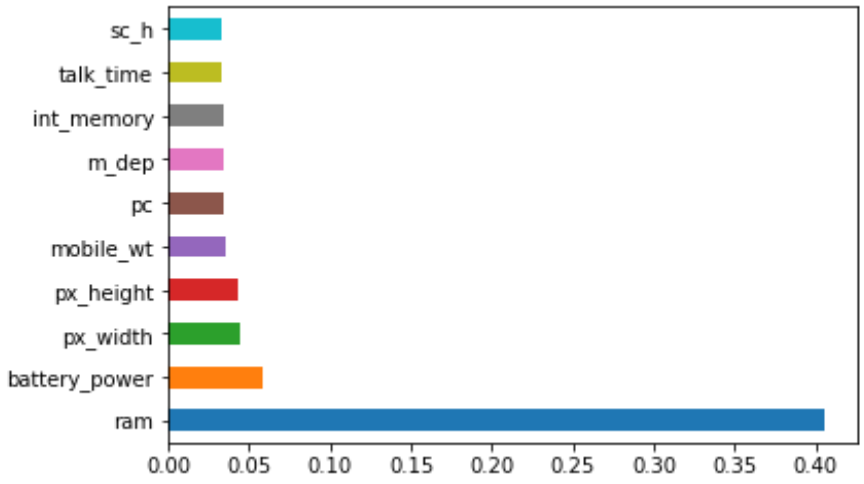


Figure 3.2 Example of feature importance showing top 10 most important features in data[8]

3. Correlation Matrix with Heatmap :

Correlation states how the features are related to each other or the target variable.Correlation can be positive (increase in one value of feature increases the value of the target variable) or negative (increase in one value of feature decreases the value of the target variable). An example of heatmap is shown in Figure 3.3.

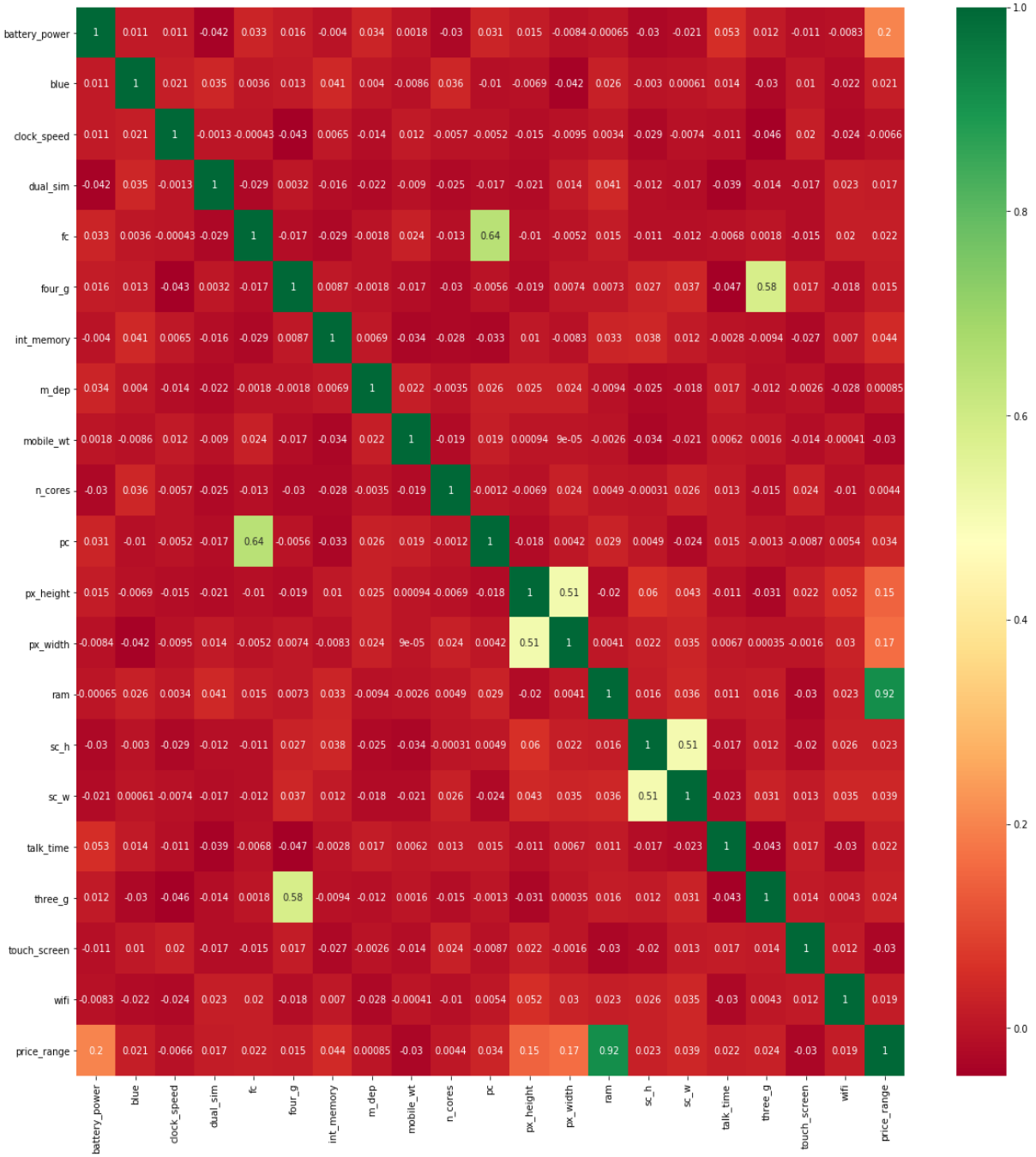


Figure 3.3 Heatmap of correlated features using seaborn library [8]

3.3 HRV Metrics and Norms:

Heart rate variability (HRV) consists of changes in the time intervals between consecutive heartbeats called interbeat intervals (IBIs). We can describe 24 h, short-term (ST, ~5 min) or brief, and ultra-short-term (UST, <5 min) HRV using time-domain, frequency-domain, and non-linear measurements [9].

3.3.1 Time-domain Measurements:

These indices of HRV quantify the amount of variability in measurements of the interbeat interval (IBI), which is the time period between successive heartbeats, observed during monitoring periods that may range from <1 min to >24 h as shown in Table 3.1.

1. SDNN :

   The SDNN is the "gold standard" for medical stratification of cardiac risk when recorded over a 24 h period [10]. SDNN values predict both morbidity and mortality. Based on 24 h monitoring, patients with SDNN values below 50 ms are classified as unhealthy, 50–100 ms have compromised health, and above 100 ms are healthy [11]. Heart attack survivors, whose 24 h measurements placed them in a higher category, had a greater probability of living during a 31-month mean follow-up period.

2. SDRR :

   SDRR measures how these intervals vary over time and is more accurate when calculated over 24 h because this longer period better represents slower processes and the cardiovascular system's response to more diverse environmental stimuli and workloads. Abnormal beats may reflect cardiac dysfunction or noise that masquerades as HRV.

3. SDANN :

   The standard deviation of the average normal-to-normal (NN) intervals for each of the 5 min segments during a 24 h recording (SDANN) is measured and reported in ms like the SDNN.

4. SDNNI:

   This measurement only estimates variability due to the factors affecting HRV within a 5-min period. It is calculated by first dividing the 24 h record into

288 5 min segments and then calculating the standard deviation of all NN intervals contained within each segment. The SDNNI is the average of these 288 values.

5. pNN50 :

   The percentage of adjacent NN intervals that differ from each other by more than 50 ms (pNN50) also requires a 2-min epoch. Researchers have proposed ultra-short-term periods of 60 s [12]. The pNN50 is closely correlated with PNS activity [13].

6. HR Max - HR Min :

   The average difference between the highest and lowest HRs during each respiratory cycle (HR Max − HR Min) is especially sensitive to the effects of respiration rate, independent of vagus nerve traffic.

7. RMSSD :

   The root mean square of successive differences between normal heartbeats (RMSSD) is obtained by first calculating each successive time difference between heartbeats in ms. Then, each of the values is squared and the result is averaged before the square root of the total is obtained. While the conventional minimum recording is 5 min, researchers have proposed ultra-short-term periods of 10 s [14], 30 s [15], and 60 s [16].

   The RMSSD reflects the beat-to-beat variance in HR and is the primary time-domain measure used to estimate the vagally mediated changes reflected in HRV [9].

8. HRV Triangular Index :

   It is a geometric measure based on 24 h recordings which calculates the integral of the density of the RR interval histogram divided by its height [10].

9. TINN :

   It is the baseline width of a histogram displaying NN intervals [10]. Like SDNN and RMSSD, contamination by only two artifacts within a 5-min segment can significantly distort its value.

Parameter	**Unit**	**Description**
SDNN	ms	Standard deviation of NN intervals

SDRR	ms	Standard deviation of RR intervals
SDANN	ms	Standard deviation of average NN intervals for 5 min segment of a 24 h HRV recording
SDNN index (SDNNI)	ms	Mean of the standard deviation of all the NN intervals for each 5 min segment of a 24 h HRV recording
pNN50	%	Percentage of successive RR intervals that differ by more than 50 ms
HR Max - HR Min	bpm	Average difference between the highest and lowest heart rates during each respiratory cycle
RMSSD	ms	Root mean square of successive RR
HRV triangular index		Integral of the density of the RR interval histogram divided by its height
TINN	ms	Baseline width of the RR interval histogram

Table 3.1 HRV Time-Domain Measures [9]

3.3.2 Frequency-domain measurements :

These indices estimate the distribution of absolute or relative power into four frequency bands. We can use Fast Fourier Transformation (FFT) or autoregressive (AR) modeling to separate HRV into its component ULF, VLF, LF, and HF rhythms that operate within different frequency ranges as shown in Table 3.2.

1. ULF Band :

   The ultra-low-frequency (ULF) band (≤0.003 Hz) requires a recording period of at least 24 h [10] and is highly correlated with the SDANN time-domain index [17].

2. VLF Band :

   The VLF band (0.0033–0.04 Hz) requires a recording period of at least 5 min, but may be best monitored over 24 h. Within a 5-min sample, there are about 0–12 complete periods of oscillation [18].

3. LF Band :

The LF band (0.04–0.15 Hz) is typically recorded over a minimum 2 min period [9]. This region was previously called the baroreceptor range because it mainly reflects baroreceptor activity during resting conditions [19].

During periods of slow respiration rates, vagal activity can easily generate oscillations in the heart rhythms that cross over into the LF band [20]. Therefore, respiratory-related efferent vagally mediated influences are particularly present in the LF band when respiration rates are below 8.5 bpm or 7 s periods [21] or when one sighs or takes a deep breath.

4. HF Band :

   The HF or respiratory band (0.15–0.40 Hz) is conventionally recorded over a minimum 1 min period. For infants and children, who breathe faster than adults, the resting range can be adjusted to 0.24–1.04 Hz [22]. The HF band reflects parasympathetic activity and is called the respiratory band because it corresponds to the HR variations related to the respiratory cycle.

   High-frequency power is highly correlated with the pNN50 and RMSSD time-domain measures [11]. HF band power may increase at night and decrease during the day [19]. Lower HF power is correlated with stress, panic, anxiety, or worry.

5. LF/HF :

   A low LF/HF ratio reflects parasympathetic dominance. This is seen when we conserve energy and engage in tend-and-befriend behaviors. A high LF/HF ratio indicates sympathetic dominance, which occurs when we engage in Fight-or-flight behaviors or parasympathetic withdrawal. Shaffer et al. [9] warned that the LF/HF ratio is controversial because different processes appear to generate 24 h and 5 min values, and these values correlate poorly.

Parameter	**Unit**	**Description**
ULF power	ms^2	Absolute power of the ultra low frequency band (<= 0.0033 Hz)
VLF power	ms^2	Absolute power of the very low frequency band (0.0033 - 0.04 Hz)
LF peak	Hz	Peak frequency of the low-frequency band (0.04-0.15 Hz)
LF power	ms^2	Absolute power of the low-frequency band

		(0.15-0.4 Hz)
LF power	nu	Relative power of the low-frequency band (0.15-0.4 Hz) in normal units
LF power	%	Relative power of the low-frequency band (0.15-0.4 Hz)
HF peak	Hz	Peak frequency of the high - frequency band (0.04-0.15 Hz)
HF power	ms^2	Absolute power of the high-frequency band (0.15-0.4 Hz)
HF power	nu	Relative power of the high-frequency band (0.15-0.4 Hz) in normal units
HF power	%	Relative power of the high-frequency band (0.15-0.4 Hz)
LF/HF	%	Ratio of LF to HF power

Table 3.2 HRV Frequency-Domain Measures [9]

3.3.3 Non-linear measurements:

These indices quantify the unpredictability and complexity of a series of IBIs, shown in Table 3.3. A Poincaré plot (return map) is graphed by plotting every R–R interval against the prior interval, creating a scatter plot. Poincaré plot analysis allows researchers to visually search for patterns buried within a time series (a sequence of values from successive measurements).

1. SD1, SD2 & SD1/SD2 :

   SD1 measures short-term HRV in ms and correlates with baroreflex sensitivity (BRS), which is the change in IBI duration per unit change in BP, and HF power. SD1 predicts diastolic BP, HR Max − HR Min, RMSSD, pNN50, SDNN, and power in the LF and HF bands, and total power during 5 min recordings [23].

   The ratio of SD1/SD2, which measures the unpredictability of the RR time series, is used to measure autonomic balance when the monitoring period is sufficiently

long and there is sympathetic activation. SD1/SD2 is correlated with the LF/HF ratio [24].

2. Approximate Entropy :

   Applied to HRV data, large ApEn values indicate low predictability of fluctuations in successive RR intervals [25]. Small ApEn values mean that the signal is regular and predictable.

3. Sample Entropy :

   Sample entropy was designed to provide a less biased and more reliable measure of signal regularity and complexity.

4. Detrended Fluctuation Analysis :

   Detrended fluctuation analysis extracts the correlations between successive RR intervals over different time scales. This analysis results in slope $\alpha 1$, which describes brief fluctuations, and slope α_2, which describes long-term fluctuations. The short-term correlations extracted using DFA reflect the baroreceptor reflex, while long-term correlations reflect the regulatory mechanisms that limit fluctuation of the beat cycle.

5. Correlation Dimension (CD,D2) :

   The CD (D2) estimates the minimum number of variables required to construct a model of system dynamics. The more variables required to predict the time series, the greater its complexity.

Parameter	**Unit**	**Description**
S	ms	Area of the ellipse which represents total HRV
SD1	ms	Poincare plot standard deviation perpendicular the line of identity
SD2	ms	Poincare plot standard deviation along the line of identity
SD1/SD2	%	Ratio of SD1 to SD2
ApEn		Approximate entropy, which measures the regularity and complexity of a time series
SampEn		Sample entropy, which measures the regularity and complexity if a time series

DFA Alpha1		Detrended fluctuation analysis, which describes short-term fluctuations
DFA Alpha2		Detrended fluctuation analysis, which describes long-term fluctuations
D2		Correlation dimension, which estimates the minimum number of variables required to construct a model of system dynamics.

Table 3.3 HRV Non-Linear Measures [9]

3.3.4 Short Term Measurement Norms:

Short-term measurement norms are based on ~5 min of HRV data (shown in Table 3.4).Because of their relative ease of recording, short-term measurements have been widely used and studied for many years, and appear to be the most commonly found source of published HRV data [26]. Short-term values are only appropriate when clients breathe at normal rates (~11–20 bpm).

Studies	**Subjects**	**Spectral analysis**	**Breathing**	**Sample**	**Position**	**Metrics**
Berkoff et al. [27]	145 elite athletes (87 M and 58 W) age 18–33	FFT	Free	2.5 min	Supine	SDNN, RMSSD, pNN50, LF (ms^2 and nu), HF (power and nu), LF/HF (% and nu), and total power
Nunan et al. [28]	21,438 healthy adults (12,960 M and 8,474 W) age ≥ 40	AR and FFT	Free/paced	Varied	Varied	RR, SDNN, RMSSD, LF (ms^2 and nu), HF (ms^2 and nu), and LF/HF

Abhishek et al. [29]	189 healthy adults (114 M and 75 W) age 16–60		Free	5 min	Supine	SDNN, RMSSD, LF (ms^2 and nu), HF (ms^2 and nu), LF/HF, and total power (ms^2)
Seppälä et al. [30]	465 prepubertal children (239 B) and 226 G age 6–8	FFT	Free	5 min	Supine	RR, HR, SDNN, RMSSD, pNN50, HTI, TINN, LF (peak, ms^2, %), HF (peak, ms^2, %), LF/HF, SD1, SD2, SD1/SD2, SampEn, D2, DFA (α1 and α2) for 5th, 25th, 50th, 75th, and 95th percentiles

Table 3.4 Short Term Measurement Norms [9]

3.3.5 24 Hour HRV Norms:

The Task Force Report [10] reported 24 hour norms for 144 healthy subjects (shown in Table 3.5) that included cutoffs for moderately depressed and highly depressed HRV and for increased risk of mortality. The authors reported 24 h time-domain measures of SDNN, SDANN, RMSSD, and the HRV HTI, and supine 5 min frequency-domain measures for LF power (LF ms^2 and nu), HF power (HF ms^2 and nu), LF/HF power, and total power (ms^2).

Studies	**Subjects**	**Metrics**
Task Force Report [10]	274 healthy subjects (202 M and 72 F), age 40–69	24 h SDNN, SDANN, RMSSD, HTI and 5 min supine LF power (ms^2 and nu), HF power (HF ms^2 and HF nu), LF/HF power, and total power
Umetani et al. [31]	260 healthy subjects (122 M and 148 W), age 10–99	SDNN, SDANN, SDNNI, RMSSD, pNN50, and HR by decade
Beckers et al. [32]	276 healthy subjects (141 M and 135 W), age 18–71	SDNN, RMSSD, and pNN50, total power, LF (ms^2 and %), HF (ms^2 and %), and LF/HF ratio, and non-linear measures, $1/f$, FD, DFA α_1 and α_2, CD, S, and LE
Bonnemeier et al. [33]	166 healthy subjects (85 M and 81 W), age 20–70	RMSSD, SDNN, SDNNI, SDANN, NN50, and HTI
Aeschbacher et al. [34]	2,079 subjects (972 M and 1,107 W), age 25–41	HR, SDNN, LF ms^2 and HF ms^2
Almeida-Santos et al. [35]	1,743 subjects (616 M and 1,127 W), age 40–100	SDNN, SDANN, SDNNI, RMSSD, and pNN50

Table 3.5 24 Hour HRV Norms [9]

Section 4: Machine Learning Techniques

Machine Learning is a branch of artificial intelligence and computer science which focuses on the use of data and algorithms to imitate the way humans learn, gradually improving the accuracy of the prediction. Machine learning identifies patterns using statistical learning and computers by unearthing boundaries in data sets. Machine Learning algorithms are commonly categorized into three different types [Figure 4.1]:

1. Supervised Learning- Input data is called training data and has a known label or result such as spam/not-spam or a stock price at a time. A model is prepared through a training process in which it is required to make predictions and is corrected when those predictions are wrong. The training process continues until the model achieves a desired level of accuracy on the training data. Example problems are classification and regression.
2. Unsupervised Learning- Input data is not labeled and does not have a known result. A model is prepared by deducing structures present in the input data. This may be to extract general rules. It may be through a mathematical process to systematically reduce redundancy, or it may be to organize data by similarity. Example problems are clustering, dimensionality reduction and association rule learning.
3. Semi-Supervised Learning- Input data is a mixture of labeled and unlabelled examples. There is a desired prediction problem but the model must learn the structures to organize the data as well as make predictions. Example problems are classification and regression.

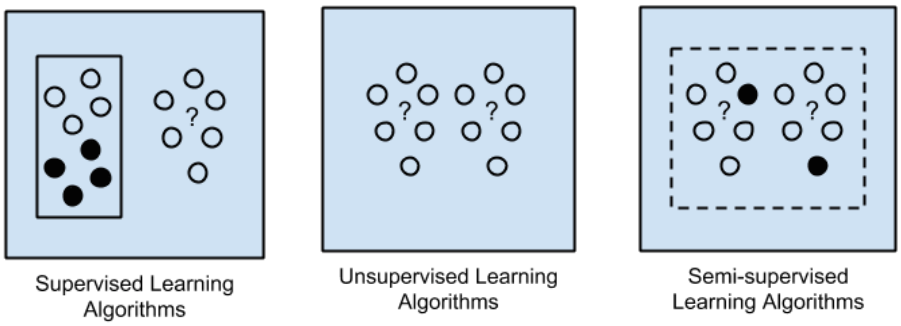


Figure 4.1: Types of Machine Learning Techniques[36]

For the purpose of this project, we dealt with the classification machine learning based techniques dealing with supervised learning. Supervised Learning is the first type of machine learning, in which labelled data is used to train the algorithms. In supervised learning, algorithms are trained using marked data, where the input and the output are

known. We input the data in the learning algorithm as a set of inputs, which is called as Features, denoted by X along with the corresponding outputs, which is indicated by Y, and the algorithm learns by comparing its actual production with correct outputs to find errors. It then modifies the model accordingly. The raw data is divided into two parts. The first part is for training the algorithm, and the other region used to test the trained algorithm [37]. The two types of Supervised Machine Learning techniques are Classification and Regression. Our project deals with the Classification problem. Classification is the process of predicting the class of given data points. Classes are sometimes called targets/ labels or categories. Classification predictive modeling is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y) [38]. A classification problem has a discrete value as its output. In our project, the state of the person will be either alert, or mildly drowsy, or in fatigue. The inputs to the classification machine learning model are the various features extracted from the raw data. For our project, we have used the data collected from a Samsung Gear S3 watch.

Some famous studies by Jongseong Gwak et al. [39] also use hybrid measures and more inputs than just physiological signals. It considered Behavioral (eg: percentage of eye closure (PERCLOS)) and Vehicle-based measures as well. According to the study, general machine learning algorithms, namely logistic regression (LR), support vector machines (SVM), the k-nearest neighbour classifier (kNN), and random forest (RF), are used for classification. The LR is a widely used algorithm for classification. It helps solve linear classification problems and binary classification problems. The SVM is also widely used for classification as a supervised learning method. It aims to maximize a value known as the margin, defined as the distance between the decision boundary and the closest training sample to the decision boundary. The SVM can efficiently perform linear classification and non-linear classification by utilizing a kernel trick. The kNN is also a commonly utilized method. It classifies the data samples based on a majority vote of their k-nearest neighbours. In addition, decision tree (DT) classification is also widely used in data mining. DT makes a model that consists of several classification trees, which predict the value of a target variable based on several input variables. RF is an ensemble of the decision tree models. It has generalization properties and runs efficiently on large databases. Furthermore, it calculates the importance of features.

The performance of the classifiers was calculated using state detection accuracy, Precision, Recall, and F1. These are defined as follows:

$$Accuracy = \frac{TP + TN}{FP + FN + TP + TN} \quad \text{---------- } Eq\ 4.1$$

$$Precision = \frac{TP}{FP + TP} \quad \text{---------- } Eq\ 4.2$$

$$Recall = \frac{TP}{FN + TP} \quad \text{---------- } Eq\ 4.3$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \quad \text{---------- } Eq\ 4.4$$

In the above formula, TP, TN, FP, and FN indicate true positives, true negatives, false positives, and false negatives respectively.

The following ML based classification algorithms were used by us:

1. K Nearest Neighbors (kNN)
2. Naive Bayes
3. Random Forest
4. Support Vector Machine

4.1 K Nearest Neighbours (KNN):

KNN, also known as K-nearest neighbour, is a supervised and pattern classification learning algorithm. It helps us find which class the new input(test value) belongs to when k nearest neighbours are chosen, and distance is calculated between them. It attempts to estimate the conditional distribution of Y given X and classify a given observation(test value) to the class with the highest estimated probability. It first identifies the k points in the training data closest to the test value and calculates the distance between all those categories (as shown in Figure 4.2). The test value will belong to the category whose distance is the least[40].

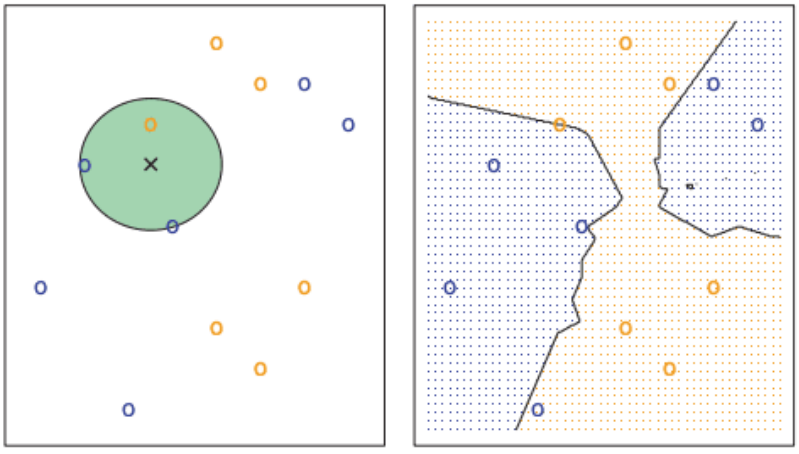


Figure 4.2 An example of the KNN algorithm classification[40]

There are different ways to calculate the distance between the categories. Some of the common methods are as follows:

1. Euclidean Method- the square root of the sum of the squared differences between a new point and an existing point across all input attributes. (We have used this for our project)
2. Hamming Distance- this is the distance between binary vectors.
3. Manhattan Distance- this is the distance between real vectors using the sum of their absolute difference.
4. Minkowski Distance- this is the generalization of Euclidean and Manhattan distance.

The algorithm's learning is different from the standard Machine Learning algorithms in that it does not learn weights from training data to predict the output. It instead uses the entire training dataset to predict the output of the test data. KNN also follows a non-parametric approach, in which there is no predefined form of the mapping function.

When KNN is used for classification, the output can be calculated as the class with the highest frequency from the K-most similar instances. Each instance, in essence, votes for its class. The class with the most votes is the final prediction. Class probabilities are calculated as the normalized frequency of samples that belong to each class in the set of K most similar instances for a new data instance. For example, in a binary classification problem (class is 0 or 1):

$$P(class = 0) = \frac{count(class=0)}{count(class=0)+count(class=1)} \text{---------- } Eq\ 4.5$$

$$P(class = 1) = \frac{count(class=1)}{count(class=0)+count(class=1)} \text{---------- } Eq\ 4.6$$

4.1.1 Pseudo Code of KNN algorithm:

The KNN Algorithm can be given as:

1. Load the data
2. Initialize K to your chosen number of neighbors
3. For each example in the data
    a. Calculate the distance between the query example and the current example from the data.
    b. Add the distance and the index of the example to an ordered collection
4. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances.
5. Pick the first K entries from the sorted collection.
6. Get the labels of the selected K entries.

7. If classification, return the mode of the K labels

The entire python implementation of the kNN algorithm has been given in the appendix B.

4.1.2 Advantages of KNN Algorithm:

1. The algorithm is simple and easy to implement.
2. It is robust to noisy training data.
3. There's no need to build a model, tune several parameters, or make additional assumptions.

4.1.3 Disadvantages of KNN Algorithm:

1. Always needs to determine the value of K which may be complex some time.
2. The computation cost is high because of calculating the distance between the data points for all the training samples.

4.1.4 Strategies Employed:

We performed the following experiments with our data:

1. First imported the `KNeighborsClassifier` function from the sklearn library.
2. Normalized the data so that plot will lead to optimal and correct results. This was done using the `StandardScaler` function from the sklearn library.
3. Apply PCA on the data to reduce the number of features.
4. Divided the dataset into training (70%) and testing (30%) partitions. We used the
5. Found an optimal value of k.
6. Passed X_train [features] and y_train [class] into the `KNeighborsClassifier` function and tested the test data, noted the accuracy.
7. Evaluated the kNN model using Confusion Matrix and Classification Report. Both functions were imported from the sklearn library.
8. Repeated the procedure for the rest of the datasets, and also for the combined dataset.

For combining the datasets, they are first normalized using the StandardScaler function, and then concatenated. We need to do this as each subject has a different mean value of the feature at a normal state. So we cannot directly compare any value for two different subjects.

4.1.5 Preparation of Data:

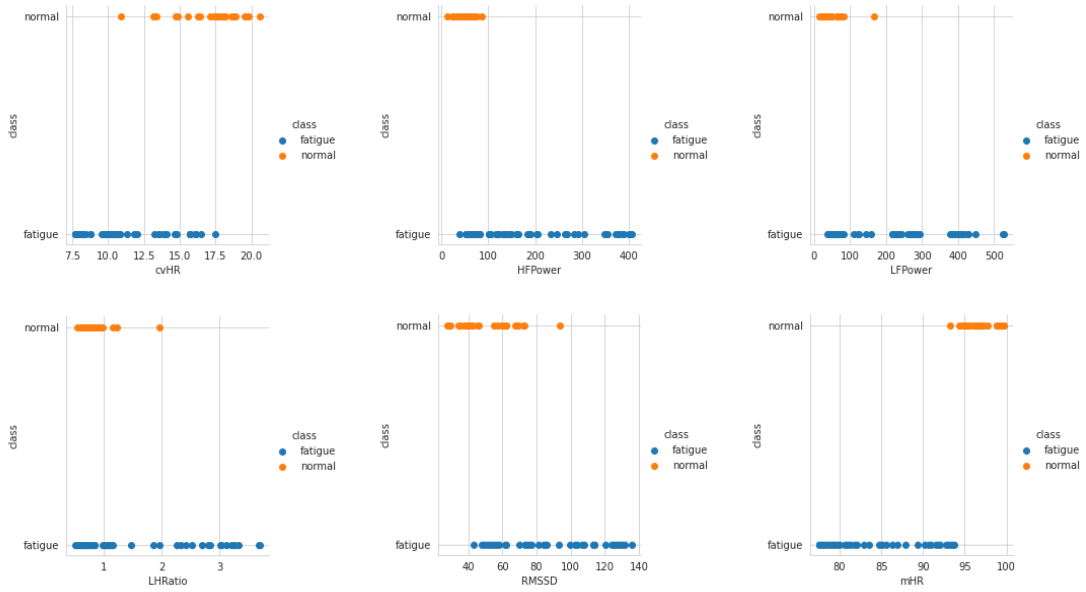
We followed the following two steps for preparing the dataset before applying the machine learning algorithms on it:

1. Data Scaling: To locate the data point in multidimensional feature space, it is helpful if all features are on the same scale. Hence normalization or standardization of data helps.
    a. The standard score of sample X, with mean μand standard deviation σ is calculated as:

$$Z = \frac{X-\mu}{\sigma} \text{---------- } Eq\ 4.7$$

2. Dimensionality Reduction: KNN may not work well if there are too many features. Hence dimensionality reduction techniques like feature selection, principal component analysis were implemented.

To get an intuitive idea about the dataset, we plotted each feature against its denoted state/class. We used the seaborn library for the plotting. The code for the same has been included in the appendix A. There were 12 features given in the data: cvHR, HF Power, mHR, pNN20, pNN30, pNN40, pNN50, LF Power, LH Ratio, RMSSD, stdHR, T Power. So there were 12 plots for each dataset. As an example, we have shown the plots for one of the dataset in Figure 4.3.



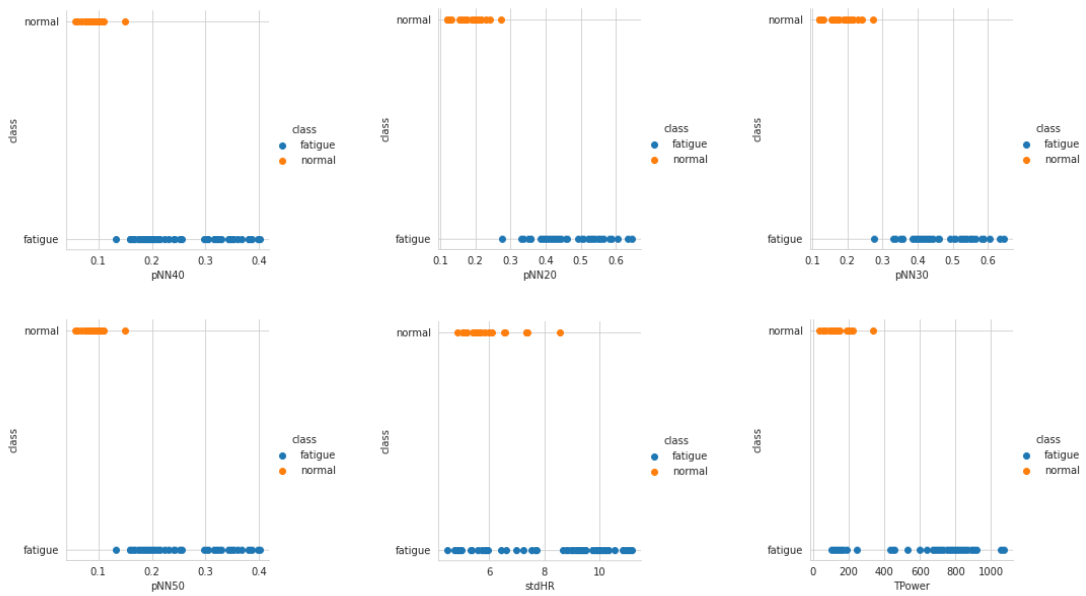


Figure 4.3: Data Visualization of features vs the class/state for one dataset

From the Figure 4.3, we can see that the classification of the state was clear/intuitive for the following features:

- cvHR, HF Power, mHR, pNN20, pNN30, pNN40, pNN50

The following features seemed ambiguous after observing the graphs (and graphs only):

- LF Power, LH Ratio, RMSSD, stdHR, T Power

We can also see that the absolute values of the features range from 0.1 for pNN20, pNN30, pNN40, and pNN50, to 1000+ for T Power. Thus normalization of data plays a huge role in kNN. This is because the algorithm works just on the distance between the sample point and the remaining K nearest neighbours. We have used the StandardScaler function from the sklearn library to satisfy this purpose.

4.1.6 Finding the Value of K:

If one uses K and there are an even number of classes (e.g. 2), it is a good idea to choose a K value with an odd number to avoid a tie. Furthermore, an even number for K is used when there is an odd number of classes. Ties can be broken consistently by expanding K by one and looking at the class of the following most similar instance in the training dataset. For our project, we had two or three classes depending on the dataset. We plotted the error of the model for each dataset against a k value ranging from 1 to 40 and then were able to find a good value of k, which minimized the error rate in the model. It varied from dataset to dataset.

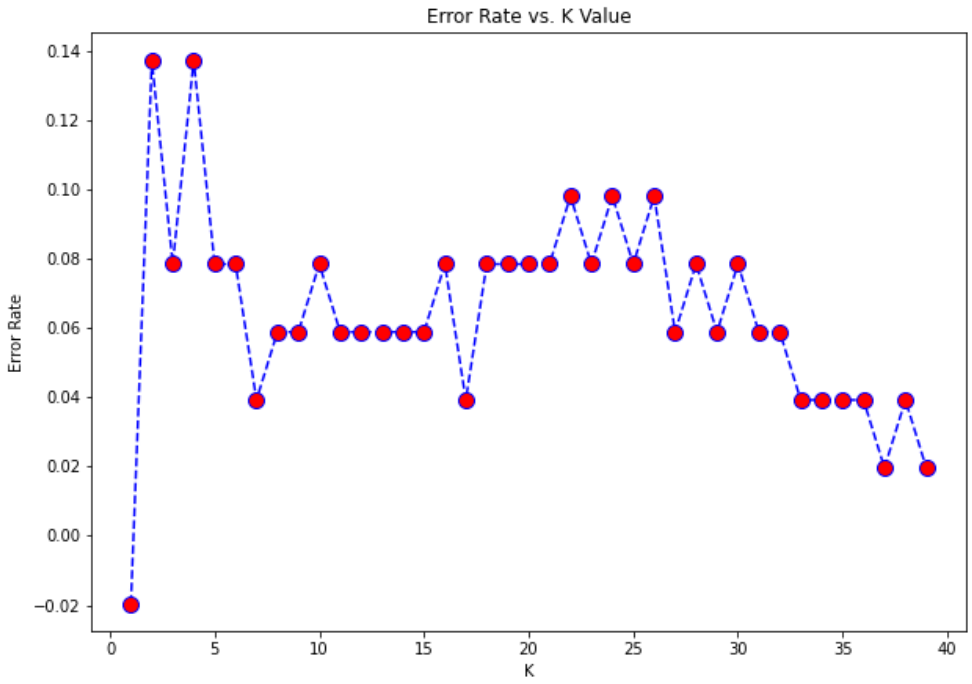


Figure 4.4: Error Rate vs K Value

From Figure 4.4 above, we can see the plot of the error of the model against the k values. Picking a low value for k can lead to outliers and give a noisy model. Since our dataset was also small, a large k value was also not suitable. For the above dataset, the k-value of 15 is the most optimal value, and it indeed gave the best accuracy, as can be seen from Table 4.1.

4.1.7 Results:

We achieved various results depending on the dataset and the corresponding k-value. We have tabulated the results for one of the datasets below:

K-Value	Accuracy
5	79.17%
10	87.50%
15	89.29%
20	83.34%
25	82.14%
30	75.00%

35	75.00%
40	71.43%

Table 4.1 K-Value vs Accuracy

From table 4.1, we can see that the k value of 15 gives a very high accuracy of 89.29%.

Next, we plot the Confusion Matrix for the model with k value 15. By definition a confusion matrix $C$is such that $C_{i,j}$ is equal to the number of observations known to be in group i and predicted to be in group j.Thus in binary classification, the count of true negatives is $C_{0,0}$, false negatives is $C_{1,0}$, true positives is $C_{1,1}$ and false positives is $C_{0,1}$. The confusion matrix for one of the kNN models can be seen in Figure 4.5.

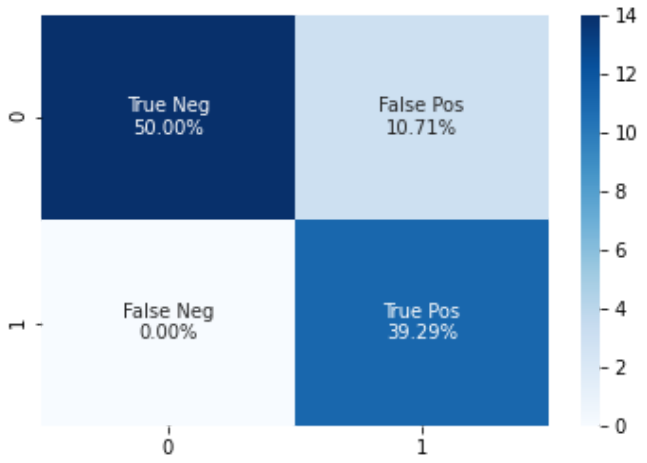


Figure 4.5: Confusion Matrix

We also formed a classification report for the model as shown in Table 4.2. We used the classification_report function from the sklearn library.

	precision	recall	f1-score	support
0 (Fatigue state)	1.00	0.82	0.90	28
1 (Normal state)	0.79	1.00	0.88	23
accuracy	-	-	0.89	51
macro avg	0.89	0.91	0.89	51
weighted avg	0.92	0.89	0.89	51

Table 4.2: Classification Report

We observe the confusion matrix [Figure 4.5] and the classification report (Table 4.2), and conclude that the model is perfect (Precision = 1.00) in estimating the Fatigue state. There were no False negatives (0.00% in the confusion matrix element $C_{1,0}$). It achieved 79% accuracy in estimating the normal state. The total accuracy of the model (k=15) was 89%.

4.2 Naïve Bayes:

The Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems. It is mainly used in text classification that includes a high-dimensional training dataset.

Naïve Bayes Classifier is one of the simplest and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions. It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.

It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the basis of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other. It is called Bayes because it depends on the principle of Bayes' Theorem.

4.2.1 Bayes' Theorem :

Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability. The formula for Bayes' Theorem is given as,

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \text{---------- } Eq\ 4.8$$

Where,

P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.

P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

P(A) is Prior Probability: Probability of hypothesis before observing the evidence.

P(B) is Marginal Probability: Probability of Evidence.

4.2.2 Advantages of Naïve Bayes Classifier :

1. Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
2. It can be used for Binary as well as Multi-class Classifications.
3. It performs well in Multi-class predictions as compared to the other Algorithms.

4.2.3 Disadvantages of Naïve Bayes Classifier :

1. Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

4.2.4 Types of Naïve Bayes Models :

1. **Gaussian**: The Gaussian model assumes that features follow a normal distribution. This means if predictors take continuous values instead of discrete, then the model assumes that these values are sampled from the Gaussian distribution.
2. **Multinomial**: The Multinomial Naïve Bayes classifier is used when the data is multinomial distributed. It is primarily used for document classification problems, it means a particular document belongs to which category such as Sports, Politics, education, etc.
   The classifier uses the frequency of words for the predictors.
3. **Bernoulli**: The Bernoulli classifier works similar to the Multinomial classifier, but the predictor variables are the independent Booleans variables. Such as if a particular word is present or not in a document. This model is also famous for document classification tasks.

Hence, for our project, the Gaussian Model works the best among these three.

4.2.5 Methodology :

The actual sklearn implementation is provided in appendix section C.

1. Data Preprocessing Step :

   Here, we prepare our data so that we can use it efficiently in our code. Firstly, we visualize the dataset [Figure 4.6] to obtain some insight on the features and labels used. For this, we use density graphs from the pyplot library.

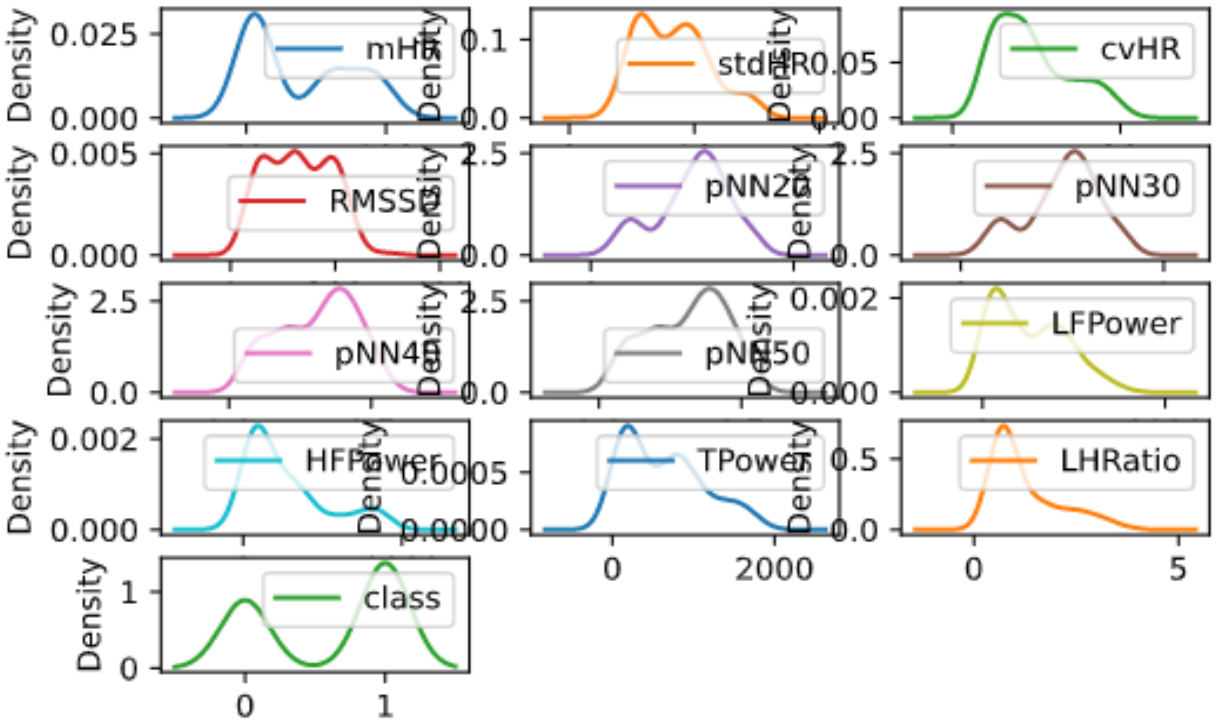


Figure 4.6: Data Visualization using Pyplot library

We import libraries such as numpy, pandas etc. and import the dataset excluding columns containing labels (i.e. excluding the column which is to be predicted). Further, the loaded dataset is divided into training and test sets using the train_test_split function.

2. Standardization : Data standardization is the process of converting data to a common format to enable users to process and analyze it. It helps establish clear, consistently defined elements and attributes, hence providing uniformity. In

standardization, mean and standard deviation are used for scaling. It is used when we want to ensure zero mean and unit standard deviation.

3. Fitting Naive Bayes to the Training set :

   Now we fit the Gaussian model to the training data. In this project, GaussianNB classifier is used.

4. Predicting the test result :

   Test result is predicted using the predict function and a new predictor variable called y_pred.

5. Test accuracy of the result :

   Accuracy is found using the accuracy_score function from sklearn.metrics library (shown in Table 4.3 and Table 4.4).

4.2.6 Results :

Test Size = 20%

	Accuracy
Without Standardization	**73.53%**
With Standardization	**76.47%**

Table 4.3 Gaussian Naive Bayes' Accuracy table for test size of 20 %

Test Size = 30%

	Accuracy
Without Standardization	**62.745%**
With Standardization	**62.75%**

Table 4.4 Gaussian Naive Bayes' Accuracy table for test size of 30 %

4.3 Random Forest:

A decision tree is a branching structure, where each branching node is a test on one of the features, where that particular feature decides which branch a training example/ test case belongs to, and the end of the branch holds a class label that is identified with that particular branch. Random forest, like its name suggests, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. Random forests are often used to measure feature importance. Random forests have a variety of applications, such as recommendation engines, image classification and feature selection. The algorithm works on the principle that a large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models [Figure 4.7]. [41]

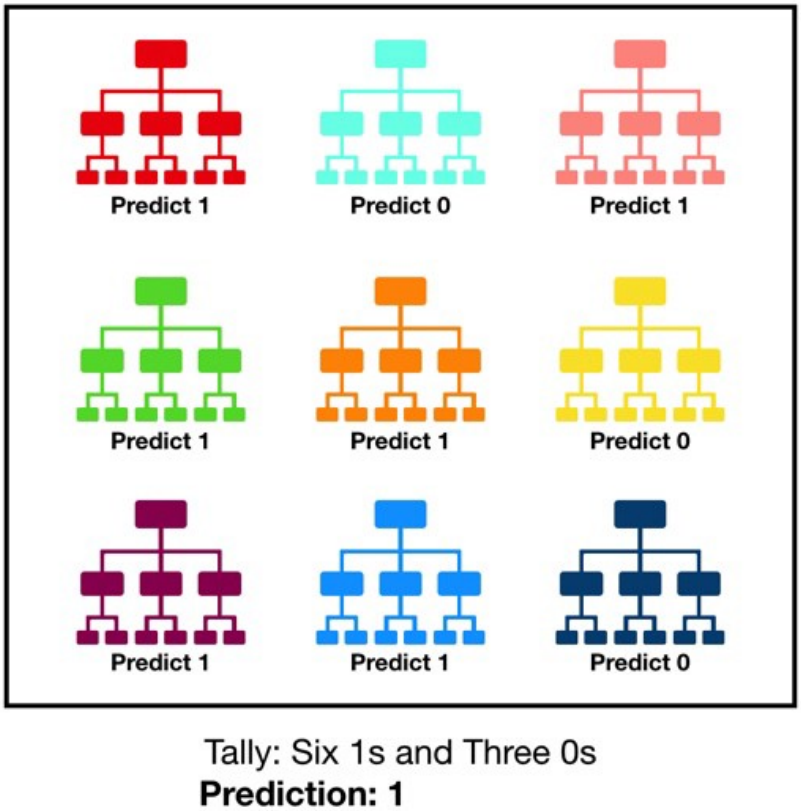


Figure 4.7 Random Forest used for classification [41]

4.3.1 Advantages of Random Forest:

1. It runs efficiently on large databases as only a part of the training data is passed onto each decision tree.
2. It gives estimates of what variables are important in the classification, very useful when there are many features as it can help in removing the features that are not helping in classifying the data.
3. It reduces the overfitting observed in single decision trees.
4. It can be used for both classification and regression problems.

4.3.2 Disadvantages of Random Forest:

1. The computational time and resources required increase significantly when using more decision trees.
2. It is hard to achieve a perfect accuracy in regression models as the average of the individual outputs of the decision trees is taken to find the final output.

4.3.3 Methodology

The two subject's data was loaded and Random Forest classifier was used in the program to give an average accuracy score for 200 iterations with randomly shuffled data (this was done to eliminate the variation in accuracy score due to shuffling that was observed when the same code was run several times). The data was randomly shuffled and divided into training sets and test sets in the ratio of 8:2 in every iteration. Then feature scaling was used on the data to give better results. The processed training set was then used in the Random Forest Classifier function to train the model and the model was used to predict the result of test set cases. Finally, the accuracy score was obtained by comparing the predicted result with the actual result of the test set. The accuracy score of 200 iterations was averaged and this was the result obtained for a particular number of decision trees. This whole code was run in a loop with the number of decision trees as the variable and the average accuracy score obtained when a specific number of decision trees was used is plotted against the number of decision trees to find the optimal number of decision trees to be used for best results. The plot is shown in Figure 4.8. The code used is given in appendix D. The variations used later are given as comments in the code.

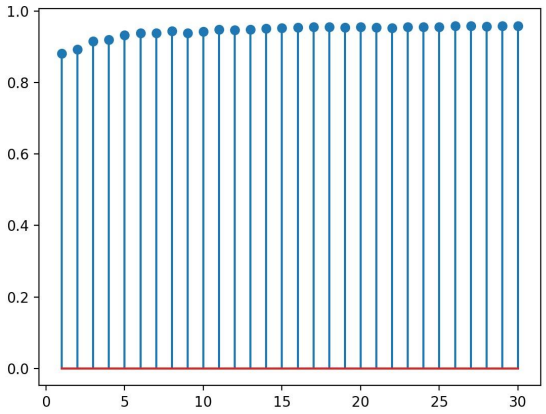


Figure 4.8 Average accuracy score obtained by the Random Forest model (y-axis) vs the number of decision trees used (x-axis) when 20% data is used as test cases

4.3.4 Results:

As evident from Figure 4.8, the model works really well with the data, achieving over 85% average accuracy with only 1 or 2 decision trees and over 95% average accuracy when 10 or more trees are used. As the computing time increases exponentially with an increasing number of decision trees, 15 trees seems to be a good choice for the model.

Figure 4.9 shows the confusion matrix and accuracy score when 15 decision trees are used and Table 4.5 has the classification report obtained by the sklearn function. 30% data was used as test sets. We can see that there is no error while classifying the 'fatigue' state, while the 'normal' state also has a very small error. A perfect precision for the fatigue state and a perfect recall for the normal state again assured that this algorithm seems to work very well for the dataset.

```
The confusion matrix is:
 [[30  2]
 [ 0 19]]

and the accuracy score is:  0.9607843137254902
```

Figure 4.9 Confusion matrix and accuracy score with 15 decision trees and 30% data used as test set

	precision	recall	f1-score	support
Fatigue state	1.00	0.94	0.97	32
Normal state	0.90	1.00	0.95	19

accuracy	-	-	0.96	51
macro avg	0.95	0.97	0.96	51
weighted avg	0.96	0.96	0.96	51

Table 4.5: Classification Report for Random Forest

Figures 4.10 and 4.11 are snippets of the output when the average accuracy score is displayed by the programme instead of plotting it. Here we can see the difference with better clarity. Figure 4.10 has the average accuracy score for the decision trees when test set size is 30% of the dataset while Figure 4.11 shows the score for 20% test set size. Initially there is a lot of randomness involved but for the number of trees ranging between 20 and 30, the result looks better when 20% data is used for the test set instead of 30%, the difference in accuracy might not seem big because of the already high accuracy but this result was obtained when the accuracy score of 200 randomly initialized tests were averaged and similar result was obtained for the number of trees ranging from 20 to 30 and thus should be valid and significant. Thus, it could be concluded that at this stage, using a bigger dataset will help improve the accuracy at least for the random forest classifier. One more thing to note here is that these average accuracy scores might look normal or even underwhelming compared to the 96% achieved with only 15 trees but these are averaged scores of 200 iterations and vary less while the earlier result varies hugely and the most common outcome was shown in Figure 4.9.

```
average accuracy for 20 trees is:  0.9492156862745089
average accuracy for 21 trees is:  0.9470588235294102
average accuracy for 22 trees is:  0.9475490196078415
average accuracy for 23 trees is:  0.9487254901960769
average accuracy for 24 trees is:  0.9483333333333317
average accuracy for 25 trees is:  0.9487254901960764
average accuracy for 26 trees is:  0.943921568627449
average accuracy for 27 trees is:  0.9463725490196064
average accuracy for 28 trees is:  0.9463725490196064
average accuracy for 29 trees is:  0.9530392156862729
average accuracy for 30 trees is:  0.9503921568627434
```

Figure 4.10 Average Accuracy when using 30% dataset as test cases

```
average accuracy for 20 trees is:  0.9552941176470573
average accuracy for 21 trees is:  0.9535294117647041
average accuracy for 22 trees is:  0.9519117647058803
average accuracy for 23 trees is:  0.9558823529411749
average accuracy for 24 trees is:  0.9558823529411746
average accuracy for 25 trees is:  0.9552941176470572
average accuracy for 26 trees is:  0.9577941176470572
average accuracy for 27 trees is:  0.9583823529411747
average accuracy for 28 trees is:  0.9560294117647041
average accuracy for 29 trees is:  0.9582352941176453
average accuracy for 30 trees is:  0.9574999999999981
```

Figure 4.11 Average Accuracy when using 20% dataset as test cases

Figure 4.12 shows the plot when the programme was run with test set size as 50% and Figure 4.13 shows the plot when it was run with only one subject's data being used as training data while the other subject's data was used as test cases. Both the iterations thus had a similar size of training set and test set, given that the dataset used the data of only two subjects. In the first case, the programme achieved over 90% average accuracy score when using more decision trees but in the second case, the accuracy score is stuck at around 50% average accuracy even when more decision trees are used. 50% accuracy in a two-state classification is obviously a very bad result and since the first case had a good accuracy, the only reason the second case did not work would be that the features extracted by PPG signals of two people vary significantly and those variations can be noticed by the ML algorithms.

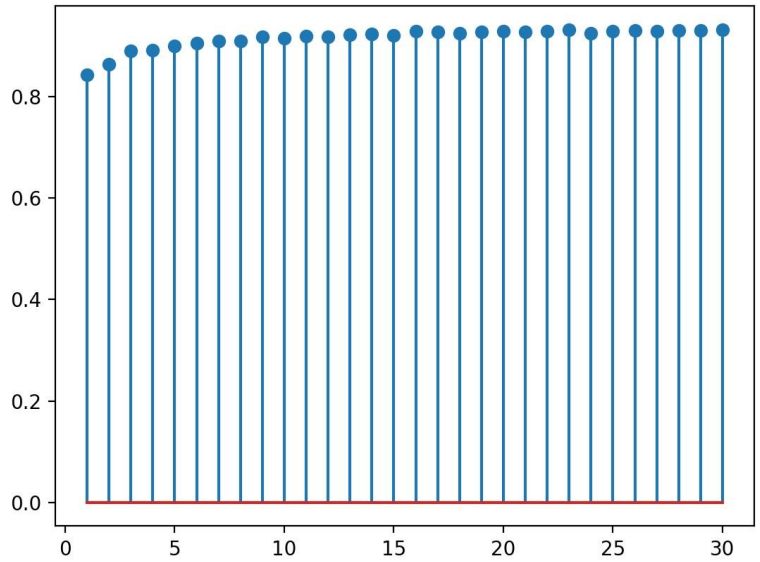


Figure 4.12 Average accuracy score plotted when test set size is 50%

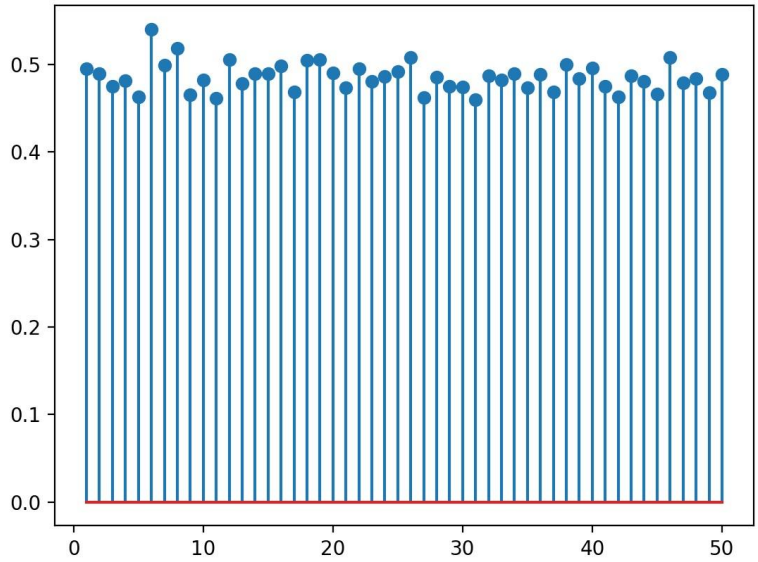


Figure 4.13 Average accuracy plotted when one subject's data was used to train the model and it was tested on the other user's data

4.4 Support Vector Machines:

Support Vector Machines are supervised machine learning models, which work on the principle of constructing a hyperplane in the N-dimensional space(N is the number of features) to perform classification of data points. SVM works on finding the best possible decision boundary to classify the data points. The hyperplane which has the maximum margin is considered to be the best possible boundary. In case there are only 2 features then the hyperplane is a line as shown in Figure 4.14.

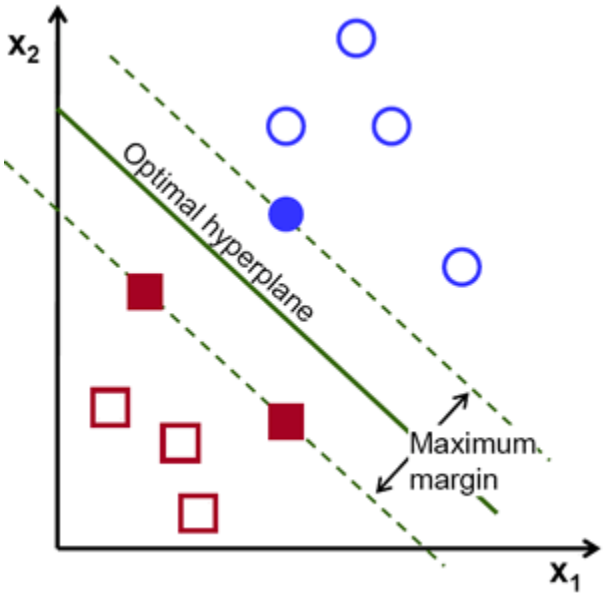


Figure 4.14: An optimal hyperplane when there are 2 features x1 and x2 [42]

As the number of features increase beyond 3, it becomes difficult to imagine the hyperplane. The data points that are used to decide the hyperplane are called support vectors. They are closest to the hyperplane and affect the orientation of the hyperplane by maximizing the margin of the hyperplane. The support vectors are chosen in such a way that they are the most similar examples between classes.

Kernel: In some cases the data might not be separable easily and in those cases we use the kernel trick. This can be done by adding another dimension. Mapping to higher dimensions can be highly computer-intensive. [43]

This is when the kernel trick comes into use wherein we don't need to go into the higher dimensional space for the computation.[44]

Types of Kernel Functions: Some of the kernel functions are as follows:

1. **Gaussian Radial Basis Function (RBF) Kernel:**

   $$K(X1,\ X2)\ =\ exp(-\frac{||X1-X2||^2}{2\sigma^2})\text{---------- } Eq\ 4.9$$

   Here, $||X_1 - X_2||$ is the Euclidean Distance and 'σ' is the variance and our hyperparameter. The width of the region of similarity is dependent on the value of σ. It is widely used.

2. **Sigmoid Kernel:**

   $$k(x,y)\ =\ tanh(\alpha x^T y\ +\ c)\text{---------- } Eq\ 4.10$$

There are two adjustable parameters in the sigmoid kernel, the slope alpha and the intercept constant c. It is also referred to as the hyperbolic tangent kernel.

3. **Polynomial Kernel:**

$k(x, y) = (\alpha x^T y + c)^d$ ---------- $Eq\ 4.11$

Adjustable parameters are the slope alpha, the constant term c and the polynomial degree d. It is mainly used when all the training data is normalized.

There are other kernels that exist too. For our project the RBF kernel works best.

4.4.1 Advantages of SVM :

1. SVMs are relatively memory efficient.
2. They are more effective in higher dimensional spaces.
3. It performs well when the number of dimensions is greater than the number of samples.

4.4.2 Disadvantages of SVM :

1. SVMs do not perform well when the data has a lot of noise.
2. They are not suitable for large datasets.

4.4.3 Applications of SVM:

1. Handwriting recognition
2. Breast cancer diagnosis
3. Protein structure prediction

4.4.4 Pseudo Code:

The actual sklearn implementation is provided in appendix E.

1. Data Preprocessing Step :

Here, we prepare our data so that we can use it efficiently in our code. We import libraries such as numpy, pandas, seaborn etc. and import the dataset.

Further, the loaded dataset is divided into training and test sets using the train_test_split function from the sklearn library keeping 20% as the test data.

2. Standardization or Feature Scaling : Data standardization is the process of converting data to a common format to enable users to process and analyze it. It helps establish clear, consistently defined elements and attributes, hence providing uniformity. In standardization, mean and standard deviation are used for scaling. It was done with the help of the StandardScaler class.
3. Fitting SVM to the Training set :

   Here, we have used the SVC(Support Vector Classifier) and used the 'rbf' kernel as it gave the best results.
4. Predicting the test set result :

   Test result is predicted using the predict function and a new predictor variable called y_pred.
5. Test accuracy of the result :

   Accuracy is found using the accuracy_score function from sklearn.metrics library.
6. Further, we obtained the confusion matrix of the results and also plotted a correlation matrix as a scatter plot as shown in Figure 4.15.

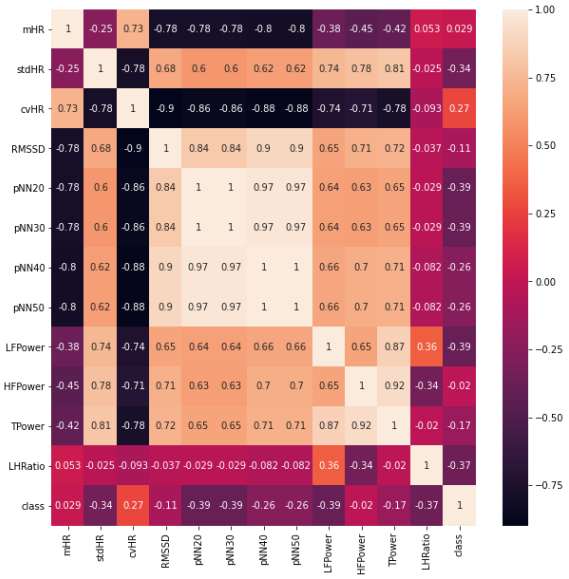


Figure4.15: Correlation matrix on the overall dataset.

4.4.5 Results:

Kernel	Accuracy
Gaussian RBF Kernel	95.29%
Sigmoid Kernel	63.40%
Linear Kernel	91.17%
Polynomial Kernel	79.60%

Table 4.6 SVM Accuracy table for test size of 20 %

From table 4.6 we can see that we get an accuracy of 95.29% for the overall data when we use the gaussian RBF kernel which is highest among all the kernels used.

Next, we plot the Confusion Matrix for the model using gaussian RBF Kernel. By definition a confusion matrix $C$is such that $C_{i,j}$ is equal to the number of observations known to be in group i and predicted to be in group j.Thus in binary classification, the count of true negatives is $C_{0,0}$, false negatives is $C_{1,0}$, true positives is $C_{1,1}$ and false positives is $C_{0,1}$. Figure 4.16 shows the confusion matrix obtained for the overall data.

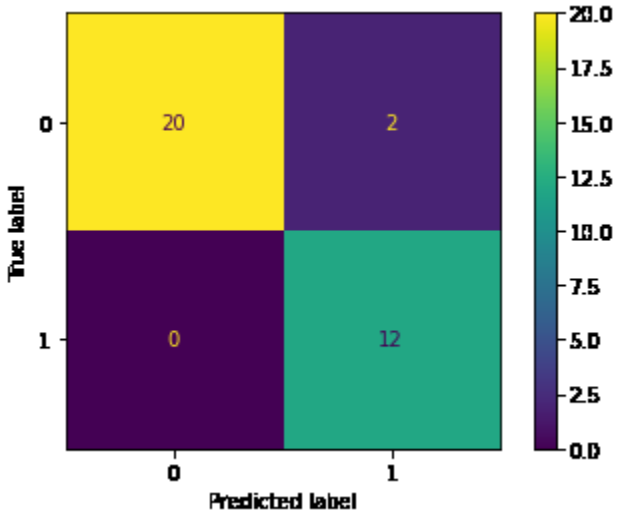


Figure4.16 Confusion matrix for overall data

We also formed a classification report for the model as shown in Table 4.7. We used the classification_report function from the sklearn library.

	precision	recall	f1-score	support

0 (Fatigue state)	1.00	0.91	0.95	22
1 (Normal state)	0.86	1.00	0.92	12
accuracy	-	-	0.94	34
macro avg	0.93	0.95	0.94	34
weighted avg	0.95	0.94	0.94	34

Table 4.7: Classification Report for SVM

We observe the confusion matrix [Figure 4.16] and the classification report (Table 4.7), and conclude that the model is perfect (Precision = 1.00) in estimating the Fatigue state. There were no False negatives (0.00% in the confusion matrix element $C_{1,0}$). It achieved 86% accuracy in estimating the normal state. Figure 4.15 shows that the feature cvHR positively impacts the normal class whereas the features pNN20, pNN30, LFPower, LHRatio negatively impacts the normal class though the impact is not very large.

Section 5: Conclusion and Future Work

This report described the need to detect a person's state using physiological signals like EEG, ECG, and PPG. The aim was to predict the state of a person while he/she does daily tasks. Since the collection of EEG and ECG signals requires large equipment, it is not feasible to collect that data. However, PPG signals were collected using a Samsung Gear S3 band worn on the watch. Signal processing reduced the noise in the signals. Filters were applied to extract time-domain and frequency-domain HR and HRV features from the PPG signals. The extracted features were used for training machine learning algorithms for person state classification.

We applied four ML-based algorithms- k Nearest Neighbours, Naive Bayes, Random Forest and Support Vector Machines. We also visualized the data in order to get an intuition about the features and the classification. The kNN algorithm was tested using a range of k values. It was observed that the k value of 15 gave the highest accuracy (89.29%).

The Naive Bayes had the least satisfying performance reaching at most 75% accuracy but it was predictable. Naive Bayes assumes that the occurrence of features are independent of each other but as we observed in Figure 4.3, a distinction boundary was not clear for more than half of the features. Thus, it was not easy to make a prediction on the basis of one feature alone. This implies that the features should be used together to form a higher-dimensional feature to predict the outcome which is not possible for Naive Bayes and hence the low accuracy. One possible solution is to remove the ambiguous and related features and try to make a prediction on the basis of the features that had a better decision boundary when plotted alone. This is a general solution that might help all the classification models.

The Random Forest Classifier worked really well in our study, achieving over 95% accuracy with 15 decision trees and is probably the best to use, given that it will not be computationally expensive to use with 15 decision trees even when handling huge amounts of data. Also, even if the number of features are increased, random forest prevents overfitting and more features can be used for more splitting and thus it can further improve the accuracy by increasing the number of decision trees.

It can be noted that a trained dataset might give very bad predictions if there is a new user, as was observed and as it is impractical to train the model everytime a new person’s state needs to be monitored, there is a need to find out ways to cancel out the variations so that the programme predicts the state of every person correctly with a pre-trained model. One possible way is to create a big enough dataset with inputs from many users so that we can ignore the variations in PPG features, as according to the

earlier experiment bigger datasets helped improving accuracy score. Another possible solution is adding more features which can be obtained from ECG and EEG signals.

Random Forest also has a great technique called feature importance that can find out if there are any features that are reducing the accuracy of the model or any useless features. It can be used to find out whether our dataset had such features, thus greatly improving the accuracy if anything significant is found. It will help classification models like Naive Bayes as well which assume every feature is independent.

Overall the highest accuracy was achieved with the help of the Support Vector Machine algorithm. An accuracy of 95.29% using the Gaussian RBF kernel which was the highest as expected. The linear kernel also gave an acceptable accuracy of 91.17%.

Further improvement in the accuracy can be achieved by improving the whole model as well. Many research papers have proposed using 3 state classification, introducing a new state, “Light Fatigue”, which might lower the risk of accident because the drivers will be alerted when they are even a little exhausted, which would not be possible with 2 state classification. Also, more features can be used, either by better feature extraction methods or using signals like EEG and/or ECG. Classification techniques can be improved by either optimizing current machine learning techniques by tuning the hyperparameters in SVM or Random Forest or by using deep learning techniques. Deep learning techniques can also be applied when the traditional methods have hit a wall, given that when there is enough data and computational power, deep learning techniques tend to produce significantly better results.

References

[1] Teplan, Michal, et al. “Spectral EEG Features of a Short Psycho-Physiological Relaxation.” *Measurement Science Review*, vol. 14, no. 4, 2014, pp. 237–242., doi:10.2478/msr-2014-0032.

[2] Vicente, José, et al. “Drowsiness Detection Using Heart Rate Variability.” Medical & Biological Engineering & Computing, vol. 54, no. 6, 2016, pp. 927–37. Crossref, doi:10.1007/s11517-015-1448-7.

[3] Awais, Muhammad, et al. “A Hybrid Approach to Detect Driver Drowsiness Utilizing Physiological Signals to Improve System Performance and Wearability.” Sensors, vol. 17, no. 9, 2017, p. 1991. Crossref, doi:10.3390/s17091991.

[4] Lal, Saroj K. L., and Ashley Craig. “A Critical Review of the Psychophysiology of Driver Fatigue.” Biological Psychology, vol. 55, no. 3, 2001, pp. 173–94. Crossref, doi:10.1016/s0301-0511(00)00085-5.

[5] Tamura, Toshiyo. “Current Progress of Photoplethysmography and SPO2 for Health Monitoring.” Biomedical Engineering Letters, vol. 9, no. 1, 2019, pp. 21–36. Crossref, doi:10.1007/s13534-019-00097-w.

[6] Allen, John. “Photoplethysmography and Its Application in Clinical Physiological Measurement.” Physiological Measurement, vol. 28, no. 3, 2007, pp. R1–39. Crossref, doi:10.1088/0967-3334/28/3/r01.

[7] Tanaka, Masaaki, et al. “Autonomic Nervous Alterations Associated With Daily Level of Fatigue.” Behavioral and Brain Functions, vol. 7, no. 1, 2011, p. 46. Crossref, doi:10.1186/1744-9081-7-46.

[8] Shaikh, Rahil. “Feature Selection Techniques in Machine Learning With Python.” Medium, Machine Learning Mastery, 28 Oct. 2018, towardsdatascience.com/feature-selection-techniques-in-machine-learning-with-python-f24e7da3f36e.

[9] Shaffer, Fred, and J. P. Ginsberg. “An Overview of Heart Rate Variability Metrics and Norms.” Frontiers in Public Health, vol. 5, 2017. Crossref, doi:10.3389/fpubh.2017.00258.

[10] Shaffer, Fred, and J P Ginsberg. “An Overview of Heart Rate Variability Metrics and Norms.” Frontiers in public health vol. 5 258. 28 Sep. 2017, doi:10.3389/fpubh.2017.00258

[11] Kleiger, R E et al. “Decreased heart rate variability and its association with increased mortality after acute myocardial infarction.” The American journal of cardiology vol. 59,4 (1987): 256-62. doi:10.1016/0002-9149(87)90795-8

[12] Wessel, Niels, et al. “Nonlinear Analysis of Complex Phenomena in Cardiological Data.” Herzschrittmachertherapie Und Elektrophysiologie, vol. 11, no. 3, 2000, pp. 159–73. Crossref, doi:10.1007/s003990070035.

[13] Umetani, Ken, et al. “Twenty-Four Hour Time Domain Heart Rate Variability and Heart Rate: Relations to Age and Gender Over Nine Decades.” Journal of the American College of Cardiology, vol. 31, no. 3, 1998, pp. 593–601. Crossref, doi:10.1016/s0735-1097(97)00554-8.

[14] Salahuddin, Lizawati et al. “Ultra short term analysis of heart rate variability for monitoring mental stress in mobile settings.” Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual International Conference vol. 2007 (2007): 4656-9. doi:10.1109/IEMBS.2007.4353378.

[15] Baek, Hyun Jae, et al. “Reliability of Ultra-Short-Term Analysis as a Surrogate of Standard 5-Min Analysis of Heart Rate Variability.” Telemedicine and E-Health, vol. 21, no. 5, 2015, pp. 404–14. Crossref, doi:10.1089/tmj.2014.0104.

[16] Esco, Michael R, and Andrew A Flatt. “Ultra-short-term heart rate variability indexes at rest and post-exercise in athletes: evaluating the agreement with accepted recommendations.” Journal of sports science & medicine vol. 13,3 535-41. 1 Sep. 2014.

[17] Bigger, J. T., et al. “Frequency Domain Measures of Heart Period Variability and Mortality After Myocardial Infarction.” Circulation, vol. 85, no. 1, 1992, pp. 164–71. Crossref, doi:10.1161/01.cir.85.1.164.

[18] Kamath, Markad, V., et al. “Methodological Aspects of Heart Rate Variability Analysis.” Heart Rate Variability (HRV) Signal Analysis: Clinical Applications, 1st ed., CRC Press, 2012, pp. 9–42.

[19] Mccraty, Rollin, and Fred Shaffer. “Heart Rate Variability: New Perspectives on Physiological Mechanisms, Assessment of Self-Regulatory Capacity, and Health Risk.” Global Advances in Health and Medicine, vol. 4, no. 1, 2015, pp. 46–61. Crossref, doi:10.7453/gahmj.2014.073.

[20] Ahmed, A. K., et al. “Respiratory Control of Heart Rate.” European Journal of Applied Physiology and Occupational Physiology, vol. 50, no. 1, 1982, pp. 95–104. Crossref, doi:10.1007/bf00952248.

[21] Tiller, W A et al. “Cardiac coherence: a new, noninvasive measure of autonomic nervous system order.” Alternative therapies in health and medicine vol. 2,1 (1996): 52-65. .

[22] Quintana, Daniel S., et al. “Resting-State High-Frequency Heart Rate Variability Is Related to Respiratory Frequency in Individuals With Severe Mental

Illness but Not Healthy Controls.” Scientific Reports, vol. 6, no. 1, 2016. Crossref, doi:10.1038/srep37212.

[23] Zerr C, Kane A, Vodopest T, Allen J, Hannan J, Cangelosi A, et al. The nonlinear index SD1 predicts diastolic blood pressure and HRV time and frequency domain measurements in healthy undergraduates [Abstract]. Appl Psychophysiol Biofeedback (2015) 40:134.10.1007/s10484-015-9282-0.

[24] Guzik, Przemysław et al. ``Correlations between the Poincaré plot and conventional heart rate variability parameters assessed during paced breathing.” The journal of physiological sciences : JPS vol. 57,1 (2007): 63-71. doi:10.2170/physiol sci.RP005506.

[25] Beckers, Frank, et al. “Approximate Entropy of Heart Rate Variability: Validation of Methods and Application in Heart Failure.” Cardiovascular Engineering: An International Journal, vol. 1, no. 4, 2001, pp. 177–82. Crossref, doi:10.1023/a:1015212328405.

[26] Armour, J. Andrew. Neurocardiology: Anatomical and Functional Principles. 1st ed., Institute of HeartMath, 2003.

[27] Berkoff, David J., et al. "Heart rate variability in elite American track-and-field athletes." Journal of strength and conditioning research 21.1 (2007): 227.

[28] NUNAN, DAVID, et al. “A Quantitative Systematic Review of Normal Values for Short-Term Heart Rate Variability in Healthy Adults.” Pacing and Clinical Electrophysiology, vol. 33, no. 11, 2010, pp. 1407–17. Crossref, doi:10.1111/j.1540-8159.2010.02841.x.

[29] Abhishekh, Hulegar A., et al. “Influence of Age and Gender on Autonomic Regulation of Heart.” Journal of Clinical Monitoring and Computing, vol. 27, no. 3, 2013, pp. 259–64. Crossref, doi:10.1007/s10877-012-9424-3.

[30] Seppälä, Santeri, et al. “Normal Values for Heart Rate Variability Parameters in Children 6–8 Years of Age: The PANIC Study.” Clinical Physiology and Functional Imaging, vol. 34, no. 4, 2013, pp. 290–96. Crossref, doi:10.1111/cpf.12096.

[31] Umetani, K et al. “Twenty-four hour time domain heart rate variability and heart rate: relations to age and gender over nine decades.” Journal of the American College of Cardiology vol. 31,3 (1998): 593-601. doi:10.1016/s0735-1097(97)00554-8.

[32] Beckers, Frank, Bart Verheyden, et al. “Aging and Nonlinear Heart Rate Control in a Healthy Population.” American Journal of Physiology-Heart and Circulatory Physiology, vol. 290, no. 6, 2006, pp. H2560–70. Crossref, doi:10.1152/ajpheart.00903.2005.

[33] Bonnemeier, Hendrik, et al. “Circadian Profile of Cardiac Autonomic Nervous Modulation in Healthy Subjects:” Journal of Cardiovascular Electrophysiology, vol. 14, no. 8, 2003, pp. 791–99. Crossref, doi:10.1046/j.1540-8167.2003.03078.x.

[34] Aeschbacher, Stefanie, et al. “Heart Rate, Heart Rate Variability and Inflammatory Biomarkers Among Young and Healthy Adults.” Annals of Medicine, vol. 49, no. 1, 2016, pp. 32–41. Crossref, doi:10.1080/07853890.2016.1226512.

[35] Almeida-Santos, Marcos Antonio, et al. “Aging, Heart Rate Variability and Patterns of Autonomic Regulation of the Heart.” Archives of Gerontology and Geriatrics, vol. 63, 2016, pp. 1–8. Crossref, doi:10.1016/j.archger.2015.11.011.

[36] Machine Learning Mastery, and Jason Brownlee. “A Tour of Machine Learning Algorithms.” Machine Learning Mastery, 12 August 2020, machinelearningmastery.com/a-tour-of-machine-learning-algorithms/.

[37] Towards Data Science, and Sidath Asiri. “Machine Learning Classifiers.” Machine Learning Classifiers, 11 June 2018, towardsdatascience.com/machine-learning-classifiers-a5cc4e1b0623.

[38] Towards Data Science, and Divyansh Dwivedi. “Machine Learning For Beginners.” Machine Learning For Beginners, 7 May 2018, towardsdatascience.com/machine-learning-for-beginners-d247a9420dab.

[39] J. Gwak, A. Hirao, and M. Shino, “An Investigation of Early Detection of Driver Drowsiness Using Ensemble Machine Learning Based on Hybrid Sensing,” Applied Sciences, vol. 10, no. 8, p. 2890, Apr. 2020, doi: 10.3390/app10082890.

[40] Singh, Ranvir. “Introduction to KNN | K-nearest neighbor classification algorithm using Examples.” Introduction to KNN, 10 June 2020, ranvir.xyz/blog/k-nearest-neighbor-algorithm-using-sklearn-distance-metric/.

[41] Yiu, Tony. “Understanding Random Forest.” Medium, Towards Data Science, 14 Aug. 2019, towardsdatascience.com/understanding-random-forest-58381e0602d2.

[42] Gandhi, Rohith. “Support Vector Machine — Introduction to Machine Learning Algorithms.” Medium, Towards Data Science, 5 July 2018, towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47.

[43] Zoltan, Czako. “SVM and Kernel SVM - Towards Data Science.” Medium, Towards Data Science, 5 May 2021, towardsdatascience.com/svm-and-kernel-svm-fed02bef1200.

[44] Team DataFlair. “Kernel Functions-Introduction to SVM Kernel & Examples.” DataFlair, Data Flair, 8 Mar. 2021, data-flair.training/blogs/svm-kernel-functions.

Appendix

A. Visualization of data using seaborn plots:

```
a = ['mHR','stdHR','cvHR','RMSSD','pNN20','pNN30','pNN40','pNN50','LFPower','HFPower','TPower','LHRatio']
for i in range(len(a)):
      sns.set_style("whitegrid")
      sns.FacetGrid(sc_df, hue="class", height=4) \
             .map(plt.scatter, i, 'class') \
             .add_legend()
plt.show()
```

B. Python Implementation of the K Nearest Neighbours Classifier:

```
# Import everything
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Create a DataFrame:
df = pd.read_csv('KNN_Dataset')

# Normalization of the data:
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaler.fit(df.drop('class', axis=1))
sc_transform = scaler.transform(df.drop('class', axis=1))
sc_df = pd.DataFrame(sc_transform)

# Use sc_df as your input features.
sc_df.head()

# Apply PCA on the data:
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
sc_df = pca.fit(pca.fit_transform(sc_df))
```

```
# Divide the data into training (70%) and testing (30%) using sklearn:
from sklearn.model_selection import train_test_split

X = sc_df
y = df['class']
X_train,   X_test,   y_train,   y_test   =   train_test_split(X,   y,
test_size=0.3)

# Use KNN and find an optimal k value:

# Initialize an array that stores the error rates.
from sklearn.neighbors import KNeighborsClassifier

error_rates = []

for a in range(1, 40):
    k = a
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)
    preds = knn.predict(X_test)
       error_rates.append(np.mean(np.asarray(y_test,  dtype=np.float64,
order ='C') - preds))

plt.figure(figsize=(10, 7))
plt.plot(range(1,40),error_rates,color='blue',      linestyle='dashed',
marker='o', markerfacecolor='red', markersize=10)
plt.title('Error Rate vs. K Value')
plt.xlabel('K')
plt.ylabel('Error Rate')
plt.savefig('ErrorRate_vs_KValue.png')

# After concluding the k value, train the kNN model and predict:
k = 20 # for example
knn = KNeighborsClassifier(n_neighbors=k)
knn.fit(X_train, y_train)
preds = knn.predict(X_test)
```

```
# Evaluate the kNN model using confusion matrix and classification report

from sklearn.metrics import confusion_matrix, classification_report

print(confusion_matrix(y_test, preds))
print(classification_report(y_test, preds))
```

C. Python Implementation of the Gaussian Naive Bayes' Classifier

```
import pandas as pd
from sklearn.metrics import accuracy_score
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn import preprocessing

#dataset has processed data of both users
data = pd.read_excel('PPG based feature extraction\\Approach 1\\dataset.xlsx')
data.info()

#dropping the column which is to be predicted
X = data.drop(['class'],axis=1)
y = data['class']


#dividing data into train and test set
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.20, shuffle=True)


#feature scaling
from sklearn.preprocessing import StandardScaler
sc= StandardScaler()
X_train= sc.fit_transform(X_train)
X_test= sc.transform(X_test)

#calling gaussian naive bayes'
```

```
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb.fit(X_train,y_train)

#predicting the test result
y_pred = gnb.predict(X_test)

#finding accuracy of the model
from sklearn.metrics import confusion_matrix,accuracy_score
cm = confusion_matrix(y_test, y_pred)
ac = accuracy_score(y_test,y_pred)
print (cm)
print (ac)
```

D. Python Implementation of the Random Forest Classifier

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

dt = pd.read_csv("test1.csv")
#test1 has the processed data of both users
#print(dt.head())
#split between i/p and o/p
X= dt.iloc[:,0:12].values
Y= dt.iloc[:,12].values

#used when only one subject's data is used for the model
#X= X[:91]
#Y= Y[:91]
#use [91:] for the other subject

esti_tree= list(range(1,31))
output=[]

for element in esti_tree:
    n=200; #taking average of accuracy score
    value=0.000;
    for i in range (n):
```

```
        #split between test and train
        from sklearn.model_selection import train_test_split
        X_train, X_test, Y_train, Y_test = train_test_split(X,Y,
         test_size= 0.3, random_state=np.random)

        #used when one subject's data is used to train the model and
        #the other subject's data is used as test set
        #X_train= X[:91]
        #Y_train= Y[:91]
        #X_test= X[91:]
        #Y_test= Y[91:]

        #feature scaling
        from sklearn.preprocessing import StandardScaler
        sc= StandardScaler()
        X_train= sc.fit_transform(X_train)
        X_test= sc.transform(X_test)

        #training
        from sklearn.ensemble import RandomForestClassifier
        classifier= RandomForestClassifier(n_estimators=element,
           random_state= np.random)
        classifier.fit(X_train,Y_train)
        Y_pred= classifier.predict(X_test)

        #accuracy result
        from sklearn.metrics import classification_report,
           confusion_matrix, accuracy_score
        value= value+accuracy_score(Y_test,Y_pred);
        #print(i,"\t")
        #print(confusion_matrix(Y_test,Y_pred))
        #print(classification_report(Y_test,Y_pred))
        #print(accuracy_score(Y_test,Y_pred))

    print("\taverage accuracy for",element,"trees is: ",
      value/float(n),"\n")
    output.append(value/float(n))

plt.stem(esti_tree,output)
plt.show()
```

```
#used for plotting the accuracy score vs the number of decision trees
```

E. Python Implementation of the Support Vector Machine

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sn
# Read and store content of an excel file
read_file = pd.read_excel ("test_overall.xlsx")

# Write the dataframe object into csv file
read_file.to_csv ("test.csv",
                  index = None,
                  header=True)

# Importing the dataset
dataset = pd.read_csv('test.csv')
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values

#Encoding categorical data
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)

# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
0.20)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Training the SVM model on the Training set
from sklearn.svm import SVC
classifier = SVC(kernel = 'rbf', random_state = 0)
```

```
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

# Predicting the Test set results
print(np.concatenate((y_pred.reshape(len(y_pred),1),
y_test.reshape(len(y_test),1)),1))

# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score,
plot_confusion_matrix, classification_report
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
print(classification_report(y_test, y_pred,))
plot_confusion_matrix(classifier, X_test, y_test)
plt.show()

#Checking correlations
datasetcopy=dataset
state={'normal':1,'fatigue':0}
datasetcopy['class']=[state[item] for item in datasetcopy['class']]
corrMatrix=datasetcopy.corr()
plt.figure(figsize=(10,10))
sn.heatmap(corrMatrix, annot=True)
plt.show()
```

Glossary

CRRI Central Road Research Institute
ECG Electrocardiography
EEG Electroencephalography
PPG Photoplethysmography
ANS Autonomous Nervous System
HRV Heart Rate Variability
SDNN Standard deviation of NN intervals
SDRR Standard deviation of RR intervals
SDANN Standard deviation of average of all the NN intervals
SDNNI Standard deviation of NN intervals index
pNN50 Percentage of successive RR intervals that differ by more than 50 ms
HR Heart Rate
RMSSD Root mean square of successive RR
TINN Baseline width of RR interval histogram
SD Standard deviation
FFT Fast Fourier transform
AR Augmented Reality
TP True Positives
TN True Negatives
FP False Positives
FN False Negatives
KNN K - Nearest Neighbours
LR Logistic Regression
DT Decision Trees
RF Random Forests
SVM Support Vector Machine
RBF Radial Basis Function