# Brain Signals Based Movement

Faculty of Computers and Information Team:
Supervisor: Dr.Sarah Nabil
BY:
Abdallah Tarek Rashad
Areej Alaa Adam
Mohamed Shawky
Bassant Mohamed
Ali Gamal

17th June, 2019

## Abstract

When a person lose his/her one of his/her tip, it would be so default thing, whatever the way. So we are trying to help those people to regain their regular life by developing a Prostheses supported and controlled by their brain. We used the machine learning to develop such solution through taking the signals of the imagine of the movement that the subject wants to do, give it to our first phase which is the preprocessing in which we pick the channels and frequencies that we will work with, then our second phase which is feature extraction in which we used the ***Common Spatial Pattern(CSP)***, ***Independent Component Analysis(ICA)*** and ***Principal component analysis(PCA)*** and our last phase which is classification in which we pass our data to our classification model and start our training using algorithms like ***Support Vector Machine(SVM)***, ***Linear Discriminant Analysis(LDA)*** and ***K-Nearest neighbour***. We succeeded to get high accuracy (will be discussed later) and a good response.

# Contents

# Chapter 1

# Introduction

## 1.1 Background

This document discuss brain computer interfacing, which the process of trying to understand and use the signals coming from the brain (EEG) in a lot of applications. Our purpose is to help paralyzed people to get back to their ordinary life. By developing a solution to replace any damaged natural tip with artificial one.

## 1.2 Motivation

Nowadays paralyzed or patients are conscious but unable to move their muscles, and this prevents them to communicate with the outside world noramlly. A typical example is patients suffering from ***Amyotrophic Lateral Sclerosis(ALS)*** an incurable, neurological disease that affects the ability to control muscles and gradually leads to paralysis. To help paralyzed persons to develop ways to communicate with external world. From this point we start that the actions are mechanical movements while thoughts are electrical impulses, thoughts are faster than actions. So, direct interaction with the world through thoughts would be faster. Then, artificial arm enables a disabled person to communicate with a computer or machine using their brain's electrical activities.

## 1.3 Problem Definition

The problem we're trying to solve here is the disability that face the people when they lose one of their natural tips, so we're trying to develop a solution that will help them get back to their normal life. Study human brain with all its functions, and it was necessary to use the headset properly and data acquisition. Selecting the channels to take the signals and feature that we need.

## 1.4 Nervous systems

Nervous systems: can be defined as organized assemblies of nerve cells as well as non nervous cells. Nerve cells, or neurons, are specialized in the generation, integration, and conduction of incoming signals from the outside world or from other neurons and deliver them to other excitable cells or to effectors such as muscle cells. Nervous systems are easily recognized in higher animals but not in the lower species, since the defining criteria are difficult to apply. [1]

### 1.4.1 Neurons

Neurons are the basic elements of the nervous system. If it put in small assemblies or clusters, they form neuronal assemblies or neuronal networks communicating with each other either chemically via synaptic junctions or electrically via tight junctions. The main characteristics of a cell are the cell body, or soma, which contains the nucleus, and a number of processes originating from the cell body, called the dendrites, which reach out to surroundings to make contacts with other cells. [2] **The brain** contains approximately 100 billion neurons [3]. Each neuron may have as many as 2000 or more connections with other neurons and may receive as many as 20; 000 inputs. Abstractly, a neuron is probably the most diverse, in terms of form and size, of all cells in the body.However, all neurons have in common the functional properties of integration, conduction and transmission of nerve impulses.

- **Cell body (or soma):** This main part has all of the necessary components of the cell, such as the nucleus (contains DNA), endoplasmic reticulum and ribosomes (for building proteins) and mitochondria (for making energy). If the cell body dies, the neuron [3]

- **Axon:** This long, cable-like projection of the cell carrie the electrochemical message (nerve impulse or action potential) along the length of the cell. Depending upon the type of neuron, axons can be covered with a thin layer of myelin, like an insulated electrical wire. Myelin is

made of fat, and it helps to speed transmission of a nerve impulse down a long axon. Myelinated neurons are typically found in the peripheral nerves (sensory and motor neurons), while non- myelinated neurons are found in the brain and spinal cord.

- **Dendrites or nerve endings:** These small,branch-like projections of the cell make connections to other cells andallow the neuron to talk with other cells or perceive the environment. Dendrites can be located on one or both ends of the cell. There are two types of dendrites, apical and basal dendrites.

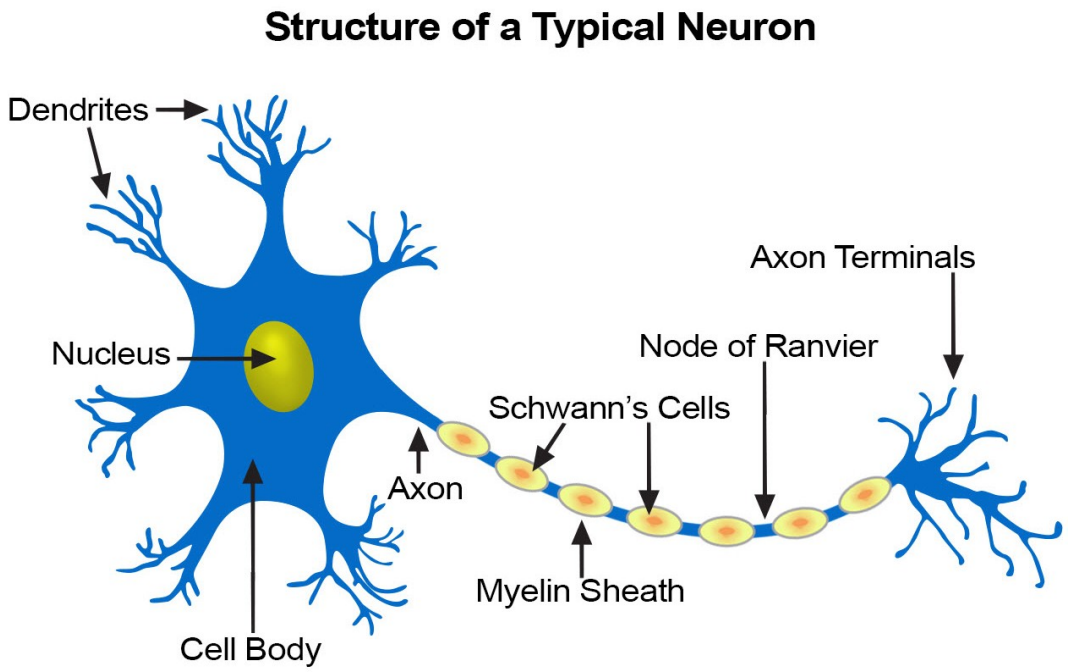



Figure 1.1: Neuron topology, each neuron is in contact through its axon and dendrites with other neurons, so that each neuron is an interconnecting segment in the network of the nervous system[3]

#### 1.4.1.1 Neurons Generation

Neurons come in many sizes. For example, a single sensory neuron from the fingertip has an axon that extends the length of the arm, while neurons within

the brain may extend only a few millimeters. Neurons have different shapes depending on what they do. Motor neurons that control muscle contractions have a cell body on one end, a long axon in the middle and dendrites on the other end; sensory neurons have dendrites on both ends, connected by a long axon with a cell body in the middle.

- **Sensory neurons:** carry signals from the outer parts of the body (periphery) into the central nervous system.
- **Motor neurons (motoneurons):** carry signals from the central nervous system to the outer parts (muscles, skin, glands) of the body.
- **Receptors:** sense the environment (chemicals, light, sound, touch) and encode this information into electrochemical messages that are transmitted by sensory neurons.
- **Interneurons:** connect various neurons within the brain and spinal cord.

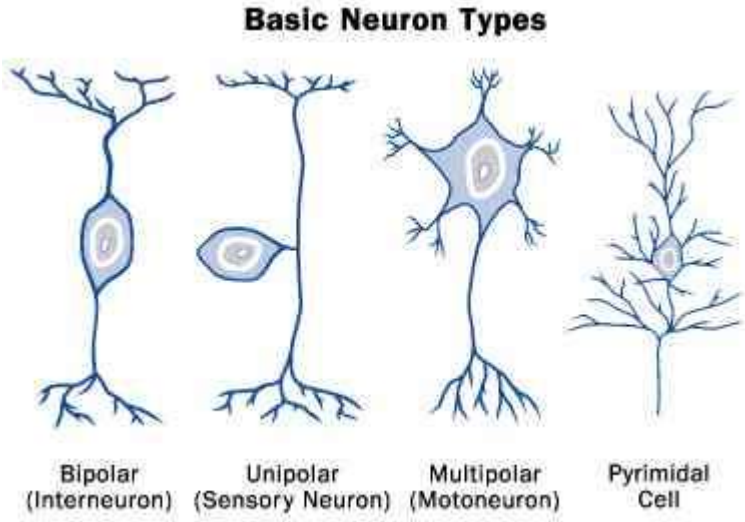



Figure 1.2: Some types of neurons: interneuron, sensory neuron, motor neuron and cortical pyramidal cell.as shown in details in

## 1.5 Functional Areas of the Brain

### 1.5.1 Brodmann Scheme

Brodmann's numbering scheme for cortical areas has been used for many years and will be introduced in this section, Projection areas. By following the course of axons entering and leaving a given cortical area, one may determine the other structures to which it is connected by afferent and efferent pathways,as shown in 1.3 [4]

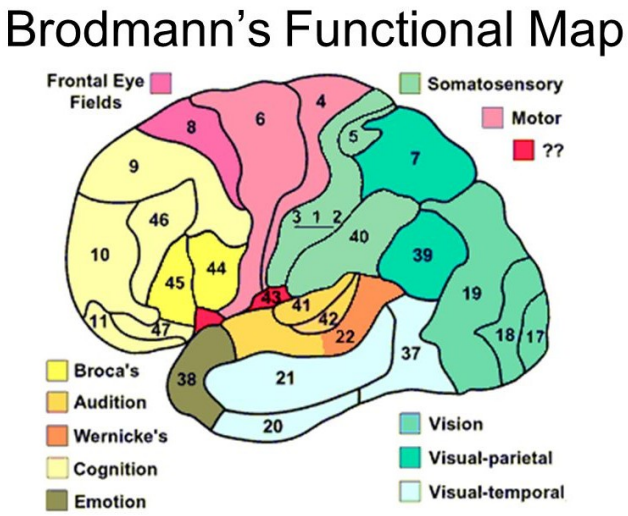



Figure 1.3: Brodmann's numbering scheme for cortical areas,it describe 50 areas neuroanatomy.

# Chapter 2

# System Architcture

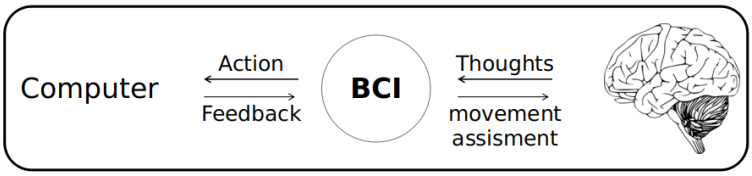



Figure 2.1:

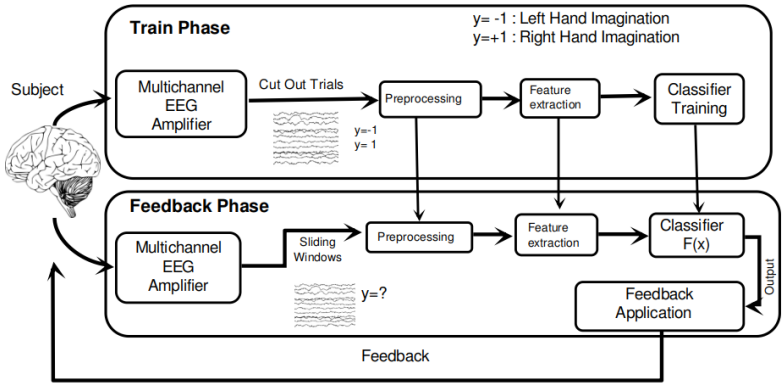



Figure 2.2:

## 2.1 Multichannel EEG Amplifier

Physiological signals characteristics have inherent challenges to their measurement, because of the presence of significant electrical noise at low frequencies in most environments. Hence, before the transmission, these signals must undergo an analog processing (amplification, filtering), especially in the case of EEG. the EEG is $\mu$V range (0.5 to 200$\mu$V) so the gain must be higher: 100000 to 1000000.

## 2.2 Processing

the preprocessing of data refers to the reading of the data, segmenting the data around interesting events such as triggers and temporal filtering.

There are largely two alternative approaches for preprocessing, which especially differ in the amount of memory required.

- The first approach is to read all data from the file into memory, apply filters, and subsequently cut the data into interesting segments (Cut of trails).
- The second approach is to first identify the interesting segments, read those segments from the data file and apply the filters to those segments only.

Preprocessing involves several steps including identifying individual trials from the dataset, filtering and artifact rejections.

Defining data segments of interest can be done

- according to a specific trigger channel
- according to your own criteria when you write your own trial function, e.g. for conditional trigger sequences, or by detecting EMG onset

## 2.3 Features Extraction

Feature extraction is the process of acquiring data about the object and Transforming it into the set of features to be classified (). When the input data to an algorithm is too large to be processed and it is suspected to be notoriously redundant (the same measurement in both feet and meters) then the input data will be transformed into a reduced representation set of features (also named features vector). If the features extracted are carefully chosen it is expected that the features set will extract the relevant information from the input data in order to perform the desired task using this reduced representation instead of the full size input. The operation can be a simple measure of physical properties of the object like length and weight. It can also be more complex like calculating the Fourier transform of the signal to find the power in a certain frequency band.

## 2.4 Classification

Classification is one of the most frequently encountered decision making tasks of human activity. A classification problem occurs when an object needs to be assigned into a predefined group or class based on a number of observed attributes related to that object (). In other words Classification, identifying to which of a set of categories a new observation belongs, on the basis of a training set of data. Many problems in business, science, industry, and medicine can be treated as classification problems. Examples include bankruptcy prediction, credit scoring, medical diagnosis, quality control, handwritten character recognition, and speech recognition.

# Chapter 3

# Previous Work

In this chapter we will preview some of the previous work in the community.

## 3.1 Introduction

There is a lot of work and development in the domain of ***Brain Computer Interfacing(BCI)***, and here we preview some glory steps in the development of such important and interesting domain:

- 1998: First (invasive, non-EEG) implant in the human brain that produces high quality signals.
- 1999: BCI is used to aid a quadriplegic for limited hand movement.
- 2002: Monkeys are trained to control a computer cursor.
- 2003: First BCI game is demonstrated to the public (BrainGate).
- 2005: Monkey brain controls a robotic arm.
- 2008: Voiceless phone calls are demonstrated (The Audeo – TI developers conference).
- 2014 Direct brain-to-brain communication achieved by transmitting EEG signals over the internet

These advancements show that BCI is a dynamic and growing field. As it is a multidisciplinary field, the evolution is affected by different factors, as new hardware, new machine learning/mathematical theories, advancements in AI and robotics, new discoveries in medicine and neuroscience, etc etc In general the field of BCI has always been limited to the academic world or medical field. Recently new startups joined the study, seeking the enhancement of human capabilities: Facebook, Neuralink, Kernel.

## 3.2 Community Work

### 3.2.1 Feature selection using regularized neighbourhood component analysis

In ***Motor Imagery(MI)*** based ***Brain–Computer Interface(BCI)*** signal analysis, **mu** and **beta** rhythms of ***Electroencephalograms(EEGs)***

are widely investigated due to their high temporal resolution and capability to define the different movement-related mental tasks separately.

However, due to the high dimensions and subject specific behavior of EEG features, there is a need for a suitable feature selection algorithm that can select the optimal features to give the best classification performance along with increased computational efficiency.

The present study proposes a feature selection algorithm based on ***Neighborhood Component Analysis(NCA)*** with modification of the regularization parameter. In the experiment, time, frequency, and phase features of the EEG are extracted using ***A Dual-Tree Complex Wavelet Transform(DTCWT)***. The proposed algorithm selects the most significant EEG features, and using these selected features, ***A Support Vector Machine(SVM)*** classifier performs the classification of MI signals.

The proposed algorithm has been validated experimentally on two public BCI datasets(BCI Competition II Dataset III and BCI Competition IV Dataset 2b).

The classification performance of the algorithm is quantified by the average accuracy and kappa coefficient, whose values are 80.7% and 0.615 respectively.

The performance of the proposed algorithm is compared with standard feature selection methods based on ***Genetic Algorithm(GA)***, ***Principal Component Analysis(PCA)***, and ***ReliefF*** and performs better than these methods.

Further, the proposed algorithm selects the lowest number of features and results in increased computational efficiency, which makes it a promising feature selection tool for an MI based BCI system. [5]

#### 3.2.1.1 Results

The average classification accuracy achieved by RNCA were 80.7%, and the best classification accuracy of 99.2% was achieved for a subject.

| | Subject ID | All Features | | | ReliefF | | | PCA | | | GA | | | RNCA | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | CA | Kappa | FS | CA | Kappa | FS | CA | Kappa | FS | CA | Kappa | FS | CA | Kappa | FS |
| Dataset 1 | BCI1 | 77.1 | 0.542 | 42 | 80.7 | 0.614 | 26 | 69.3 | 0.586 | 11 | **81.4** | **0.628** | 7 | 80.7 | 0.614 | 11 |
| Dataset 2 | B0101T | 84.2 | 0.684 | 42 | 88.3 | 0.766 | 16 | 72.2 | 0.444 | 15 | 91.7 | 0.834 | 19 | **93.3** | **0.866** | 6 |
| | B0102T | 82.5 | 0.650 | 42 | 91.7 | 0.834 | 10 | 65 | 0.300 | 12 | 91.7 | 0.834 | 8 | **92.5** | **0.850** | 7 |
| | B0201T | 62.5 | 0.250 | 42 | 64.2 | 0.284 | 22 | 57.5 | 0.150 | 12 | 72.5 | 0.450 | 8 | **73.3** | **0.466** | 16 |
| | B0202T | 60.0 | 0.200 | 42 | 62.5 | 0.250 | 12 | 56.7 | 0.134 | 15 | 70.8 | 0.416 | 8 | **73.3** | **0.466** | 7 |
| | B0301T | 58.3 | 0.166 | 42 | 64.2 | 0.284 | 13 | 54.2 | 0.084 | 11 | 66.7 | 0.334 | 4 | **67.5** | **0.350** | 7 |
| | B0302T | 49.2 | −0.016 | 42 | 51.7 | 0.234 | 33 | 50.8 | 0.016 | 10 | **54.2** | **0.084** | 4 | 49.2 | −0.016 | 3 |
| | B0401T | 87.5 | 0.75 | 42 | 80.0 | 0.600 | 10 | 97.5 | 0.950 | 12 | 95.8 | 0.916 | 25 | **99.2** | **0.984** | 4 |
| | B0402T | 92.1 | 0.842 | 42 | 91.4 | 0.828 | 38 | 77.1 | 0.542 | 10 | **94.3** | **0.886** | 3 | **94.3** | **0.886** | 5 |
| | B0501T | 75.8 | 0.516 | 42 | 90.0 | 0.800 | 11 | 64.2 | 0.284 | 9 | **91.7** | **0.834** | 6 | **91.7** | **0.834** | 6 |
| | B0502T | **73.6** | **0.472** | 42 | 72.1 | 0.442 | 13 | 70.0 | 0.600 | 15 | 70.7 | 0.414 | 7 | **73.6** | **0.472** | 8 |
| | B0601T | 79.2 | 0.584 | 42 | 80.8 | 0.616 | 30 | 60.0 | 0.200 | 11 | 86.7 | 0.734 | 8 | **94.2** | **0.884** | 3 |
| | B0602T | 76.7 | 0.534 | 42 | 85.0 | 0.700 | 12 | 60.8 | 0.216 | 9 | 87.5 | 0.750 | 5 | **88.3** | **0.766** | 2 |
| | B0701T | 67.5 | 0.350 | 42 | 65.8 | 0.316 | 19 | 59.2 | 0.184 | 11 | 65.8 | 0.316 | 5 | **70.8** | **0.416** | 11 |
| | B0702T | 61.7 | 0.234 | 42 | **65.8** | **0.316** | 23 | 50.8 | 0.016 | 9 | 63.3 | 0.266 | 11 | 64.2 | 0.284 | 9 |
| | B0801T | 67.5 | 0.350 | 42 | 73.1 | 0.462 | 13 | 62.5 | 0.250 | 13 | **78.8** | **0.576** | 5 | 78.1 | 0.562 | 2 |
| | B0802T | 75.8 | 0.516 | 42 | 65.8 | 0.316 | 7 | 70.0 | 0.400 | 10 | 78.3 | 0.566 | 6 | **80.8** | **0.616** | 6 |
| | B0901T | 67.5 | 0.35 | 42 | 85.0 | 0.700 | 8 | 66.7 | 0.334 | 11 | 75.0 | 0.500 | 6 | **88.3** | **0.766** | 2 |
| | B0902T | 73.3 | 0.466 | 42 | 81.7 | 0.634 | 25 | 57.5 | 0.150 | 13 | **83.3** | **0.666** | 8 | 81.7 | 0.634 | 11 |
| Mean Values | | 72.2 | 0.444 | 42 | 75.7 | 0.526 | 18 | 64.3 | 0.307 | 11.52 | 78.9 | 0.579 | 8 | **80.7** | **0.615** | 6.6 |

Figure 3.1: This figure shows the accuracy over different feature extractors

### 3.2.2 Reduce Calibration Time in Motor Imagery

One major challenge in the development of a brain–computer interface is to reduce calibration time or completely eliminate it.
To address this problem, existing approaches use covariance matrices of ***electroencephalography(EEG)*** trials as descriptors for decoding BCI but do not consider the geometry of the covariance matrices, which lies in the space of ***Symmetric Positive Definite(SPD)*** matrices.
This inevitably limits their performance. They focus on reducing calibration time by introducing SPD based classification approach. However, SPD-based classification has limited applicability in small training sets because the dimensionality of covariance matrices is large in proportion to the number of trials.
To overcome this drawback, our paper proposes a new framework that transforms SPD matrices in lower dimension through spatial filter regularized by

prior information of EEG channels. The efficacy of the proposed approach was validated on the small sample scenario through Dataset IVa from BCI Competition III.

The proposed approach achieved mean accuracy of 86.13% and mean kappa of 0.72 on Dataset IVa. The proposed method outperformed other approaches in existing studies on Dataset IVa. Finally, to ensure the robustness of the proposed method, they evaluated it on Dataset IIIa from BCI Competition III and Dataset IIa from BCI Competition IV. The proposed method achieved mean accuracy 92.22% and 81.21% on Dataset IIIa and Dataset IIa, respectively. [6]

#### 3.2.2.1 Results

| Parameters | Dataset IVa | | | | | Dataset IIIa | | | | Dataset IIa | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | aa | al | av | aw | ay | k3b | k6b | l1b | A01 | A02 | A03 | A04 | A05 | A06 | A07 | A08 | A09 |
| $\alpha$ | $10^{-1}$ | $10^{-4}$ | $10^{-5}$ | $10^{-5}$ | $10^{-1}$ | $10^{-4}$ | $10^{-3}$ | $10^{-10}$ | $10^{-4}$ | $10^{-2}$ | $10^{-3}$ | $10^{-3}$ | $10^{-4}$ | $10^{-2}$ | $10^{-3}$ | $10^{-10}$ | $10^{-10}$ |
| r | 0.06 | all r values | 0.07 | 0.07 | 0.08 | 0.1 | 0.08 | 0.01 | 0.06 | 0.07 | 0.05 | 0.09 | 0.07 | 0.04 | 0.06 | 0.01 | 0.01 |

Figure 3.2: This figure shows the accuracy over different datasets

### 3.2.3 Transfer Learning of BCI Using CUR Algorithm

In order to have a high performing BCI classification, the training set must contain variations of high-quality subjects which are discriminative.
Variations will also drive transferability of training data for generalization purposes. However, if the test subject is unique from the training set variations, BCI performance may suffer. Previously, this problem was solved by introducing transfer learning in the context of spatial filtering on small training set by creating high quality variations within training subjects.
In this study however, it was discovered that transfer learning can also be used to compress the training data into an optimal compact size while improving training data performance.
The transfer learning framework proposed was on motor imagery BCI-EEG

using CUR matrix decomposition algorithm which decomposes data into two components; C and UR which is each subject's EEG signal and common matrix derived from historical EEG data, respectively. The method is considered transfer learning process because it utilizes historical data as common matrix for the classification purposes. This framework is implemented in the BCI system along with ***Common Spatial Pattern(CSP)*** as features extractor and ***Extreme Learning Machine(ELM)*** as classifier and this combination exhibits an increase of accuracy to up to 26% with 83% training database compression. [7]

#### 3.2.3.1 Results

| Subjects | CSP | ACSP | RCSP | TL-CSP | | | |
|---|---|---|---|---|---|---|---|
| | | | | Mean | MeanBP | Diff | Dev |
| 1a | 0.66 (0.88) | 0.63 (0.92) | 0.61 (0.78) | 0.85 (0.95) | 0.84 (0.94) | 0.85 (0.94) | 0.84 (0.93) |
| 1b | 0.49 (0.88) | 0.59 (0.93) | 0.56 (0.83) | 0.76 (0.86) | 0.77 (0.88) | 0.81 (0.89) | 0.76 (0.86) |
| 1c | 0.77 (0.91) | 0.75 (0.97) | 0.74 (0.78) | 0.87 (0.95) | 0.86 (0.95) | 0.87 (0.95) | 0.85 (0.94) |
| 1d | 0.91 (0.96) | 0.92 (1.00) | 0.93 (0.99) | 0.97 (0.99) | 0.97 (0.99) | 0.97 (0.99) | 0.98 (0.99) |
| 1e | 0.92 (0.99) | 0.94 (0.99) | 0.96 (0.99) | 0.96 (0.99) | 0.98 (0.99) | 0.98 (0.99) | 0.97 (0.99) |
| 1f | 0.52 (0.88) | 0.54 (0.95) | 0.59 (0.86) | 0.92 (0.92) | 0.92 (0.93) | 0.92 (0.93) | 0.92 (0.93) |
| 1 g | 0.64 (0.87) | 0.69 (0.13) | 0.89 (0.92) | 0.92 (0.94) | 0.92 (0.94) | 0.92 (0.93) | 0.93 (0.94) |
| Average | 0.71 (0.92) | 0.72 (0.95) | 0.76 (0.88) | 0.89 (0.94) | 0.89 (0.94) | 0.90 (0.95) | 0.89 (0.94) |

Figure 3.3: This figure shows the effect of the transfer learning

### 3.2.4 Separated channel convolutional neural network

In this paper, They introduce an end-to-end deep learning framework to realize the training free ***Motor Imagery(MI)*** BCI systems.
Specifically, they employ the ***Common Space Pattern(CSP)*** extracted from ***electroencephalography(EEG)*** as the handcrafted feature. Instead of log-energy, they use the multi-channel series in CSP space to retain the temporal information.
Then they propose a separated channel convolutional network, here termed SCCN, to encode the multi-channel data.
Finally, the encoded features are concatenated and fed into a recognition network to perform the final MI task recognition.
We compared the results of the deep model with classical machine learning algorithms, such as ***K-Nearest Neighbors(KNN)***, ***Logistics Regression(LR)***, ***Linear Discriminant Analysis(LDA)***, and ***Support Vector Machine(SVM)***. Moreover, the quantitative analysis was evaluated on our dataset and the BCI competition IV-2b dataset. [8]

#### 3.2.4.1 Results

The results have shown that our proposed model can improve the accuracy of EEG based MI classification (2–13% improvement for our dataset and 2–15% improvement for BCI competition IV-2b dataset) in comparison with traditional methods under the training free condition.

### 3.2.5 High Classification Accuracy of a Motor Imagery Based Brain-Computer Interface

The current work presents a comparative evaluation of the MI-based BCI control accuracy between stroke patients and healthy subjects. Five patients who had a stroke that affected the motor system participated in the current study, and were trained across 10–24 sessions lasting about 1 h each with the recoveriX system.
The participants' EEG data were classified while they imagined left or right hand movements, and real-time feedback was provided on a monitor. If the

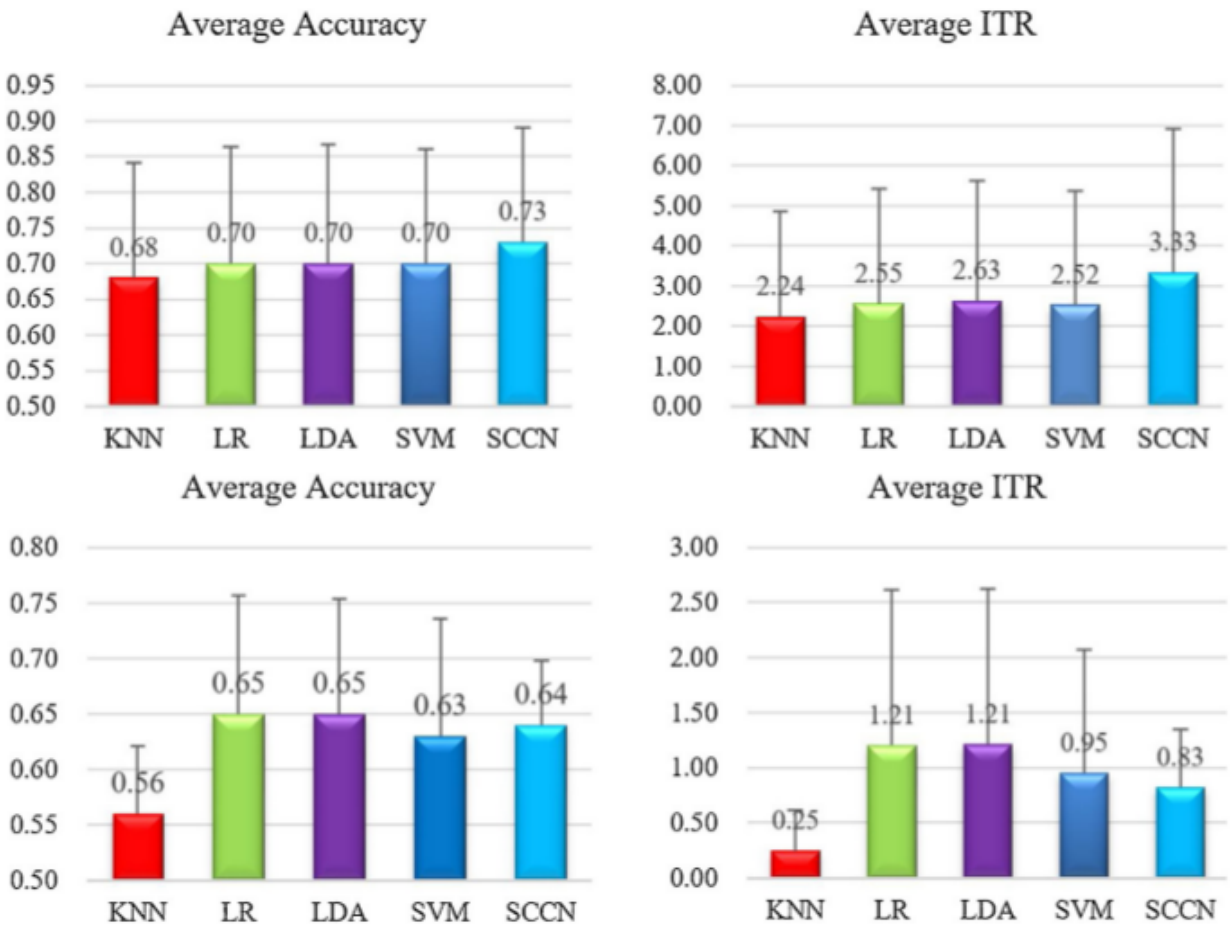



Figure 3.4: This figure shows the effect of the deep learning frameworks

correct imagination was detected, the FES was also activated to move the left or right hand.

The grand average mean accuracy was 87.4% for all patients and sessions. All patients were able to achieve at least one session with a maximum accuracy above 96%. Both the mean accuracy and the maximum accuracy were surprisingly high and above results seen with healthy controls in prior studies. mportantly, the study showed that stroke patients can control a MI BCI system with high accuracy relative to healthy persons. This may occur because these patients are highly motivated to participate in a study to improve their motor functions. Participants often reported early in the training of motor

improvements and this caused additional motivation. However, it also reflects the efficacy of combining motor imagination, seeing continuous bar feedback, and real hand movement that also activates the tactile and proprioceptive systems.
Results also suggested that motor function could improve even if classification accuracy did not, and suggest other new questions to explore in future work. Future studies will also be done with a first-person view 3D avatar to provide improved feedback and thereby increase each patients' sense of engagement.
The present results suggest that stroke patients can control a BCI with high accuracy even with the lesioned hemisphere, despite damage to motor areas and other potential problems (such as diminished attention, concentration, or motivation). Interestingly, the grand average accuracy of all 5 patients is 87.4% and the mean maximum accuracy is 96.9% across all subjects and sessions (10 to 24). [9]

#### 3.2.5.1 Results

All patients reached an average accuracy above or equal to 82.7% during the training, and patients P2 and P5 were above 90%. Every patient attained accuracy above or equal to 96.2% in at least one session. P1, P3, and P5 improved their accuracy with training, P2 consistently exhibited high accuracy, and P4's accuracy actually declined during training, though it did not vary much overall.

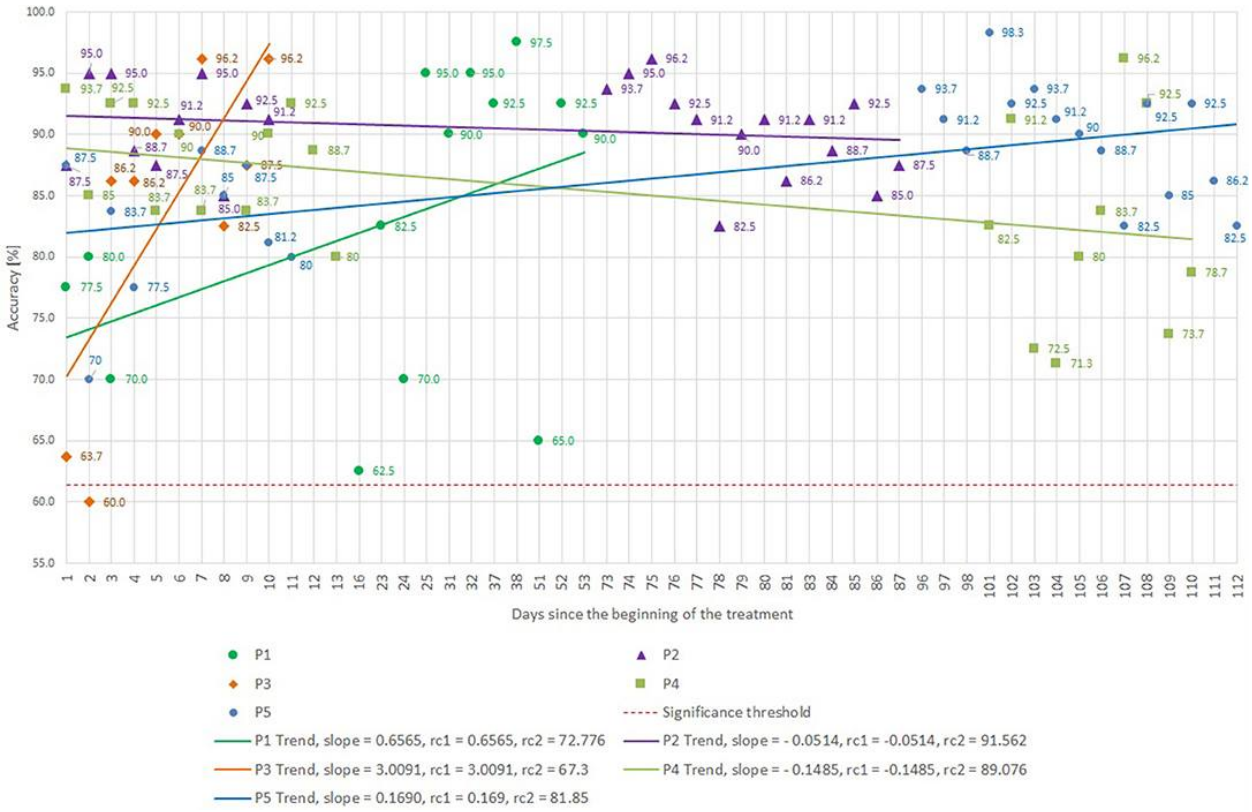



Figure 3.5: Figure presents BCI control accuracy scores for all five patients, based on the online results (not post-hoccross-validated data) achieved while recording runs five and six (the data from runs 1–4 was used to set up the CSP filter and to train the LDA), and the trend-lines of the accuracies in each patient. Patients P1, P2, P4, and P5 started with an accuracy above 70%. P3 began at only 61%, which is below the significance threshold of 61.4%, but improved rapidly. The threshold for significant accuracy depends on the total number of trials (Billinger et al., 2013; Yuan et al., 2013), and they developed the significance threshold of 61.4% for each run based on MATLAB's binofit function, which uses the Clopper-Pearson method for calculating the confidence intervals (Cruse et al., 2011).

| Patient | Minimum accuracy [%] | Average accuracy [%] | Maximum accuracy [%] | Accuracy improvement | Motor improvement |
|---|---|---|---|---|---|
| P1 | 62.5 | 82.7 | 97.5 | Yes | Yes |
| P2 | 82.5 | 90.3 | 96.2 | No | Yes |
| P3 | 60 | 83.7 | 96.2 | Yes | Yes |
| P4 | 71.3 | 85.6 | 96.2 | No | Yes |
| P5 | 70 | 94.5 | 98.3 | Yes | Yes |
| Total | 69.3 | 87.4 | 96.9 | | |

Figure 3.6: This figure shows the Minimum, average, and maximum accuracies

| Sessions | Paretic hand Time [s]/dropped PEGs | | | Healthy hand Time [s]/dropped PEGs | | |
|---|---|---|---|---|---|---|
| | P1 | P2 | P5 | P1 | P2 | P5 |
| 0 (baseline) | 52/1 | 65/- | | 46/- | 31/- | |
| 3 | 52/2 | 54/- | | 46/- | 32/- | |
| 6 | 45/- | 45/- | | 40/1 | 32/- | |
| 9 | 40/- | 42/- | 90/1 | 35/- | 31/- | 26/- |
| 12 | 40/- | 42/- | 77/- | 32/- | 31/- | 26/- |
| 15 | 38/- | 38/- | 94/- | 33/- | 29/- | 26/- |
| 18 | | 34/- | 60/- | | 29/- | 25/- |
| 21 | | 30/- | 61/- | | 29/- | 25/- |
| 24 | | 30/- | 52/- | | 29/- | 26/- |
| Total time | 14 | 35 | 38 | 13 | 2 | 0 |

Figure 3.7: This figure shows the results of 9-hole PEG tests for P1, P2, and P5. The first line in the table, numbered as session zero, represents the baseline performance for that patient. P1's 9-hole PEG test time improved for both healthy and paretic hands. P2 and P5 improved with the paretic hand. The "total time improvement" row at the bottom of Table 3 shows that all three participants required less time to perform the 9-HPT with the paretic hand after training. P1 and P2 may have exhibited a "ceiling effect," as their performance with the paretic hand became close (P1) or equal (P2) to the non-paretic hand.

# Chapter 4

# Data Acquisition

## 4.1 Introduction

We have defined data acquisition as the method used for capturing EEG signals from the surface of the scalp. Different methods are available to record EEG. Because of physiological variability and different size and shapes of human heads, it is hard to develop a perfect system to exactly access specific regions of the brain. The International 10-20 System standardize EEG electrodes placement and is widely adopted for EEG studies[10].

## 4.2 International 10-20 System

The ***International Federation of Societies for Electroencephalography and Clinical Neurophysiology*** has recommended the conventional electrode setting (also called 10-20) 1 for 21 electrodes (excluding the earlobe electrodes)[11].

## 4.3 Electrodes

The EEG is recorded with electrodes, which are placed on the scalp. Electrodes are small plates, which conduct electricity. They provide the electrical contact between the skin and the EEG recording apparatus by transforming the ionic current on the skin to the electrical current in the wires. To improve the stability of the signal, the outer layer of the skin called stratum corneum should be at least partly removed under the electrode. Electrolyte gel is applied between the electrode and the skin in order to provide good electrical contact. Usually small metalplate electrodes are used in the EEG recording[12].

## 4.4 Equipments(Emotiv Epoc+)

The award-winning EMOTIV EPOC+ is designed for scalable and contextual human brain research and provides access to professional grade brain data with a quick and easy to use design.


Figure 4.1: Emotiv Epoc+

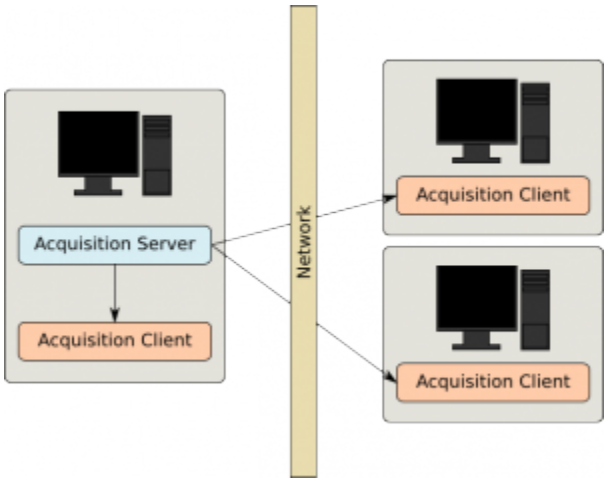
We have also used Research Edition Headset from Emotiv. This headset gives EEG signal with sampling frequency of 128 Hz.
Emotiv Epoc+ headset is a wireless device that can be connected to a computer using their Research Software Development Kits (SDK).
**EEG sensors** 14 channels: AF3, F7, F3, FC5, T7, P7, O1, O2, P8, T8, FC6, F4, F8, AF4 2 references: CMS/DRL references at P3/P4; left/right mastoid process alternative

## 4.5 OpenVIBE

The purpose of OpenViBE is to get data from the acquisition device through the **Acquisition Server** and then send it to one or more clients. This client is usually, for now, the OpenViBE designer. The Acquisition Server and the clients (Designers) can be either on the same machine or different machines on the same network, or any combination of these.
The following diagram explains the possibilities: The main OpenViBE ap-


Figure 4.2: EEG sensors Position

plication fields are medical (assistance to disabled people, real-time biofeedback, neurofeedback, real-time diagnosis), multimedia (virtual reality, video games), robotics and all other application fields related to brain-computer interfaces and real-time neurosciences. OpenViBE users can either be programmers or people not familiar with programming. This includes medical doctors, video game developers, researchers in signal processing or robotics, etc. Interface The user interface of OpenVibe is easy to use for creating BCI scenarios and saving them for later use, to access and to manipulate. OpenVibe is the first library of functions written in C++ of this type developed by INRIA - Institut national de recherche en informatique et automatique (France) - it can be integrated and applied quickly and easily .


Acquisition Server
Acquisition Client
Network
Acquisition Client
Acquisition Client

# Part I

# Proposed System

# Chapter 5

# Features Extraction

## 5.1 Introduction

Feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing. A characteristic of these large data sets is a large number of variables that require a lot of computing resources to process. Feature extraction is the name for methods that combine variables into features, effectively reducing the amount of data that must be processed, while still accurately and completely describing the original data set.

### 5.1.1 Purpose of Features Extraction

The process of feature extraction is useful when you need to reduce the number of resources needed for processing without losing important or relevant information. Feature extraction can also reduce the amount of redundant data for a given analysis.

## 5.2 Methods of Features Extraction

- ..
  - ..
- ..

## 5.3 Common Spatial Pattern(CSP)

CSP is a technique to analyze multichannel data based on recordings from two classes (conditions). CSP yields a data-driven supervised decomposition of the signal parameterized by a matrix $W \in \mathbb{R}^{C\times C}$ (C being the number of channels) that projects the signal $x(t) \in \mathbb{R}^C$ in the original sensor space to $\mathbf{x}_{CSP}(t) \in \mathbb{R}^C$ which lives in the surrogate sensor space [13] , as follows:

$$x_{CSP}(t) = W^T x(t) \tag{5.1}$$

the purpos of CSP is to find a spatial filter to maximize variance for one class and minimize variance for another class at the same time to improve classification.

### 5.3.1 CSP Algorithm

1. Calculate the covariance matrix:
   Calculate the covariance matrix for each class

$$\Sigma^{+} \in \mathbb{R}^{C\times C} \quad (For\, class\, 1) \tag{5.2}$$

$$\Sigma^{-} \in \mathbb{R}^{C\times C} \quad (For\, class\, 2) \tag{5.3}$$

$$\Sigma^{(c)} = \frac{1}{|I_c|}\sum_{i\in I_c} X_i X_i^T \quad (c \in \{+,-\}) \tag{5.4}$$

   Where $I_c\,(c \in \{+,-\}$ is the set of idices corresponding to trials belonging to each class and $|I|$ denotes the size of a set $I$

2. Calculate the simultaneous diagonalization:
   finding eigenvectors common to both covariance matrices

$$W^T\Sigma^{+}W = \Lambda^{+} \tag{5.5}$$

$$W^T\Sigma^{-}W = \Lambda^{-} \tag{5.6}$$

   $W$ consisting of the generelized eigenvectors $w_j (j = 1, ..., C)$

$$\Sigma^{+}w = \lambda\Sigma^{-}w \tag{5.7}$$

   $\lambda_j^c = w_j^T\Sigma^c w_j$ being the corresponding diagonal element of $\Lambda^c\,(c \in \{+,-\})$ while $\lambda$ in 5.7 equal $\frac{\lambda_j^+}{\lambda_j^-}$

3. Calculate spatial filter $W$:

$$W = \underset{W \in \mathbb{R}^{C \times C}}{\operatorname{argmax}} \left\{ \frac{W^T \Sigma^+ W}{W^T \Sigma^- W} \right\} \tag{5.8}$$

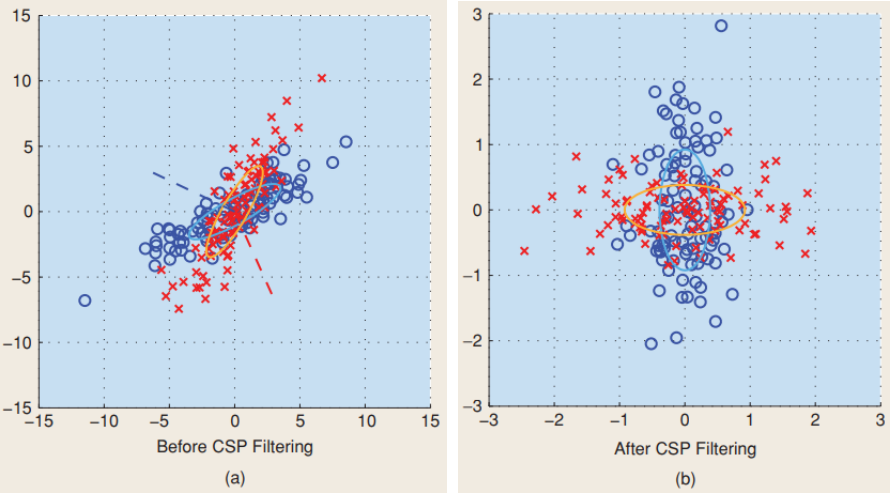



Figure 5.1: A toy example of CSP filtering in 2-D. Two sets of samples marked by red crosses and blue circles are drawn from two Gaussian distributions. In (a), the distribution of samples before filtering is shown. Two ellipses show the estimated covariances and dashed lines show the direction of CSP projections $w_j (j = 1, 2)$. In (b), the distribution of samples after the filtering is shown. Note that both classes are uncorrelated at the same time; the horizontal (vertical) axis gives the largest variance in the red (blue) class and the smallest in the blue (red) class, respectively.

## 5.4 Independent Component Analysis (ICA)

The concept of independent component analysis (ICA) lies in the fact that the signals may be decomposed into their constituent independent components. In places where the combined source signals can be assumed independent from each other this concept plays a crucial role in separation and denoising the signals[10].

A measure of independency may easily be described to evaluate the independence of the decomposed components. Generally, considering the multichannel signal as $y(n)$ and the constituent signal components as $y_i(n)$, the $y_i(n)$ are independent if

$$p_Y(y(n)) = \prod_{i=1}^{m} p_y(y_i(n)) \quad \forall n \tag{5.9}$$

where $p(Y)$ is the joint probability distribution, $p_y(y_i(n))$ are the marginal distributions and $m$ is the number of independent components.

### 5.4.1 Blind Source Separation (BSS)

An important application of ICA is in blind source separation (BSS). BSS is an approach to estimate and recover the independent source signals using only the information of their mixtures observed at the recording channels. Due to its variety of applications BSS has attracted much attention recently. BSS of acoustic signals is often referred to as the 'cocktail party problem' [14], which means separation of individual sounds from a number of recordings in an uncontrolled environment such as a cocktail party. Figure 5.2 illustrates the BSS concept.

### 5.4.2 ICA Algorithm

$\mathbf{x}$ the random vector whose elements are the mixtures $x_1, ..., x_n$, and likewise by $\mathbf{s}$ the random vector with elements $s_1, ..., s_n$. Let us denote by $\mathbf{A}$ the matrix with elements $a_{ij}$

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{5.10}$$

All we observe is the random vector $\mathbf{x}$, and we must estimate both $\mathbf{A}$ and $\mathbf{s}$ using it.
after estimating the matrix $\mathbf{A}$, we can compute its inverse, say $\mathbf{W}$, and obtain the independent component simply by:

$$\mathbf{s} = \mathbf{W}\mathbf{x} \tag{5.11}$$

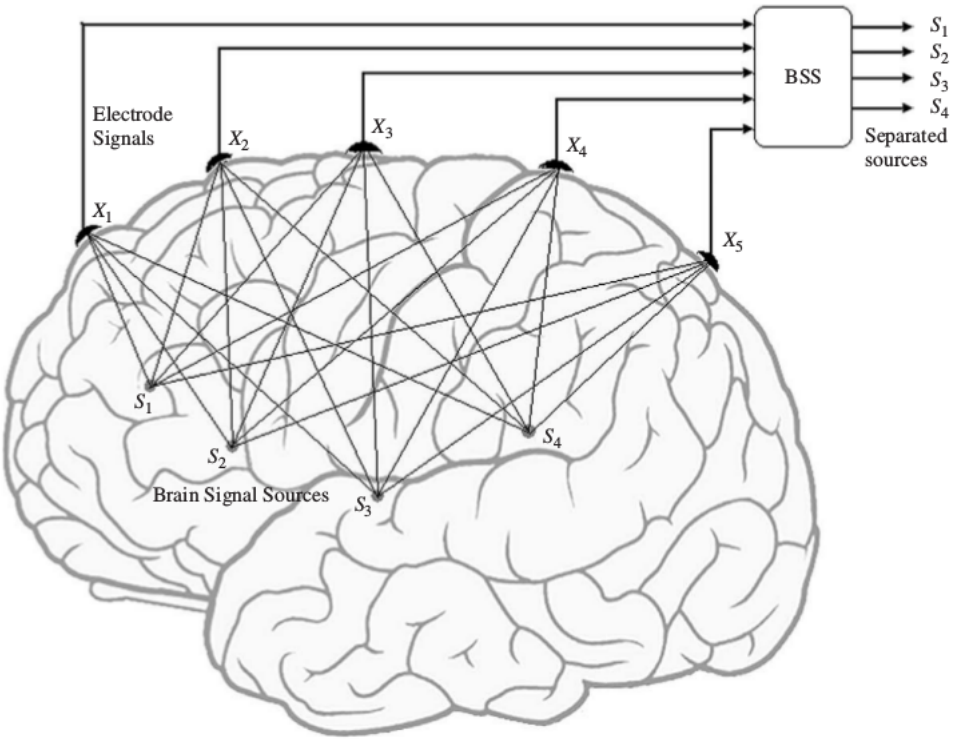



Figure 5.2: BSS concept; mixing and blind separation of the EEG signals

The singular value decomposition (SVD) is a linear algebra technique that provides a method for dividing A into several simpler pieces. For any matrix SVD states that

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T \tag{5.12}$$

We estimate A and its inverse W by recovering each piece of the decomposition individually:

$$\mathbf{W} = \mathbf{A}^{-1} = \mathbf{V}\Sigma^{-1}\mathbf{U}^T \tag{5.13}$$

Calculate the covariance of the data:

$$\begin{aligned} \mathbf{x}\mathbf{x}^T &= (\mathbf{A}\mathbf{s})(\mathbf{A}\mathbf{s})^T \\ &= (\mathbf{U}\Sigma\mathbf{V}^T\mathbf{s})(\mathbf{U}\Sigma\mathbf{V}^T\mathbf{s})^T \\ &= \mathbf{U}\Sigma\mathbf{V}^T(\mathbf{s}\mathbf{s}^T)\mathbf{V}\Sigma\mathbf{U}^T \end{aligned} \tag{5.14}$$

assume that the covariance of the sources $\mathbf{s}$ is *whitened*, or equivalently $\mathbf{s}\mathbf{s}^T = \mathbf{I}$ where $\mathbf{I}$ is the identity matrix and a property of orthogonal matrices

$\mathbf{V}\mathbf{V}^T = \mathbf{I}$

$$\mathbf{x}\mathbf{x}^T = \mathbf{U}\Sigma^2\mathbf{U}^T \tag{5.15}$$

we can solve this equation by eigendecomposition form it is familiar in linear algebra by $\mathbf{U}$ whose columns are the eigenvectors of the covariance of $\mathbf{x}$ and $\Sigma^2$ is a diagonal matrix of associated eigenvalues

## 5.5 Principal Component Analysis (PCA)

Principal component analysis (PCA) has proven to be an exceedingly popular technique for dimensionality reduction and is discussed at length in most texts on multivariate analysis.
Its many application areas include data compression, image analysis, visualization, pattern recognition, regression and time series prediction[15].

### 5.5.1 PCA Algorithm

1. **Subtract the mean:**
   you have to subtract the mean from each of the data dimensions. The mean subtracted is the average across each dimension. So, all the $x$ values have $\bar{x}$ (the mean of the $x$ values of all the data points) subtracted, and all the $y$ values have $\bar{y}$ subtracted from them. This produces a data set whose mean is zero

2. **Calculate the covariance matrix:**
   Covariance is a measure of the extent to which corresponding elements from two sets of ordered data move in the same direction. We use the following formula to compute covariance.

   $$Cov(X,Y) = \frac{1}{N}\sum(X_i - \bar{X})(Y_i - \bar{Y}) \tag{5.16}$$

   Variance and covariance are often displayed together in a variance-covariance matrix, (aka, a covariance matrix). The variances appear

along the diagonal and covariances appear in the off-diagonal elements, as shown below.

$$\Sigma = \frac{1}{N} \sum X_i X_i^T \tag{5.17}$$

3. **Calculate the eigen vectors $V$ and eigen values $D$ of the covariance matrix:**
A scalar $\lambda$ is called an eigenvalue of the $n \times n$ matrix $\Sigma$ is there is a nontrivial solution $\mathbf{x}$ of $\Sigma\mathbf{x} = \lambda\mathbf{x}$. Such an $\mathbf{x}$ is called an eigenvector corresponding to the eigenvalue $\lambda$.
If $v_1, v_2, \ldots, v_n$ are linearly independent eigenvectors of $\Sigma$ and $\lambda_1, \lambda_2, \ldots, \lambda_n$ are their corresponding eigenvalues, then $\Sigma = \mathbf{VDV}^{-1}$ , where

$$V = [v_1 \; v_2 \; \ldots \; v_n] \tag{5.18}$$

$$D = \begin{bmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \ldots & \lambda_n \end{bmatrix} \tag{5.19}$$

4. **Reduce dimensionality and form feature vector:**
The eigenvector with the highest eigenvalue is the principal component of the data set.
Once eigenvectors are found from the covariance matrix, the next step is to order them by eigenvalue, highest to lowest. This gives the components in order of significance.
Now, if you'd like, you can decide to ignore the components of lesser significance.You do lose some information, but if the eigenvalues are small, you don't lose much
When the $\lambda_i$'s are sorted in descending order, the proportion of variance explained by the $r$ principal components is:

$$\frac{\sum_{i=1}^{r} \lambda_i}{\sum_{i=1}^{m} \lambda_i} = \frac{\lambda_1 + \lambda_2 + \cdots + \lambda_r}{\lambda_1 + \lambda_2 + \cdots + \lambda_n} \tag{5.20}$$

- If the dimensions are highly correlated, there will be a small number of eigenvectors with large eigenvalues and $r$ will be much smaller than $n$.

- If the dimensions are not correlated, $r$ will be as large as $n$ and PCA does not help.

5. **Derive the new data:**

$$\textbf{FinalData} = \textbf{RowFeatureVector} \times \textbf{RowZeroMeanData} \quad (5.21)$$

**RowFeatureVector** is the matrix with the eigenvectors in the columns transposed so that the eigenvectors are now in the rows, with the most significant eigenvector at the top.

**RowZeroMeanData** is the mean-adjusted data transposed, i.e., the data items are in each column, with each row holding a separate dimension.

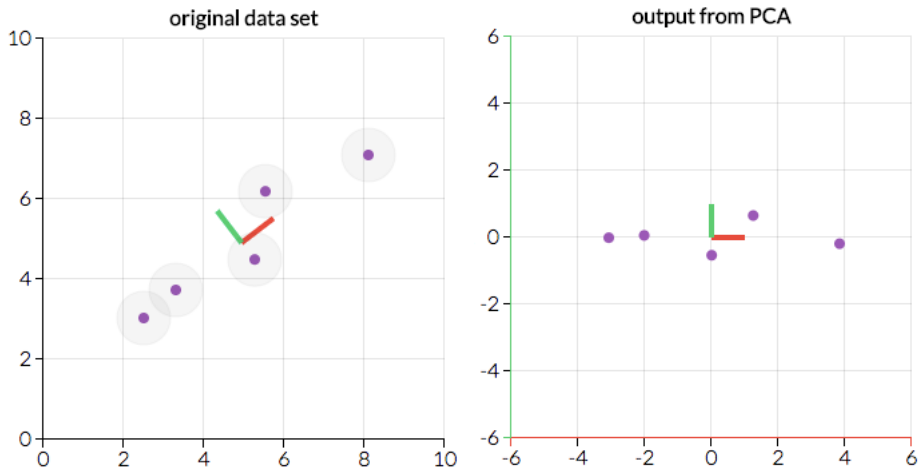



Figure 5.3: the data before and after PCA

# Chapter 6

# Classification

## 6.1 Introduction

Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/labels or categories. Classification predictive modeling is the task of approximating a mapping function ($f$) from input variables ($X$) to discrete output variables ($y$).
Classification belongs to the category of supervised learning where the targets also provided with the input data. There are many applications in classification in many domains such as in credit approval, medical diagnosis, target marketing etc.

There are two types of learners in classification as lazy learners and eager learners.

1. **Lazy learners**
   Lazy learners simply store the training data and wait until a testing data appear. When it does, classification is conducted based on the most related data in the stored training data. Compared to eager learners, lazy learners have less training time but more time in predicting like k-nearest neighbor.

2. **Eager learners**
   Eager learners construct a classification model based on the given training data before receiving data for classification. It must be able to commit to a single hypothesis that covers the entire instance space. Due to the model construction, eager learners take a long time for train and less time to predict like Decision Tree and Naive Bayes.

### 6.1.1 Regions Of Classification

In our Classification phase we parted the 64 channels in three regions:

1. **Region 1**
   We picked some of the 64 channels based on the mean accuracy over those channels which are:
   Fc5, Fc3, C5, C3, Cp5, Cp3, Af7, F7, F5, F8, Ft7, T7, T9, Tp7, P7.

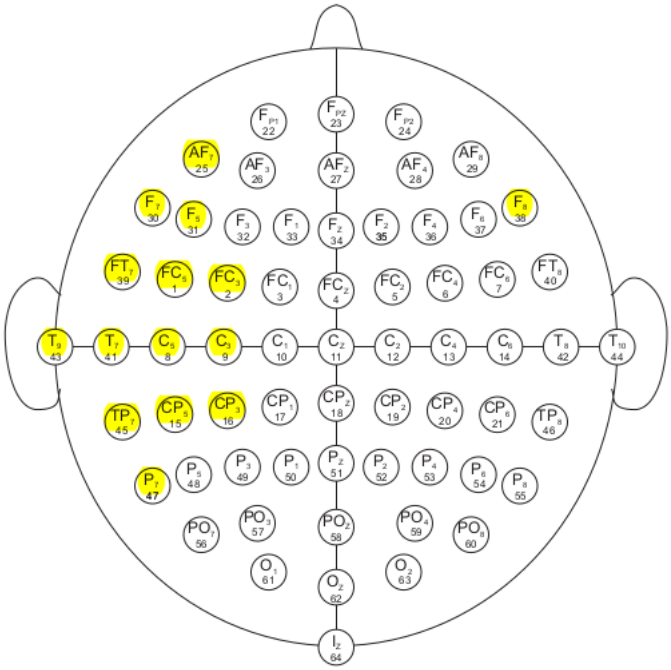


Figure 6.1: This figure shows annotations for region 2.

2. **Region 2**
   We picked some of the 64 channels based on the mean accuracy over those channels which are:
   C5, F7, Ft7, T7, T9, Tp7.

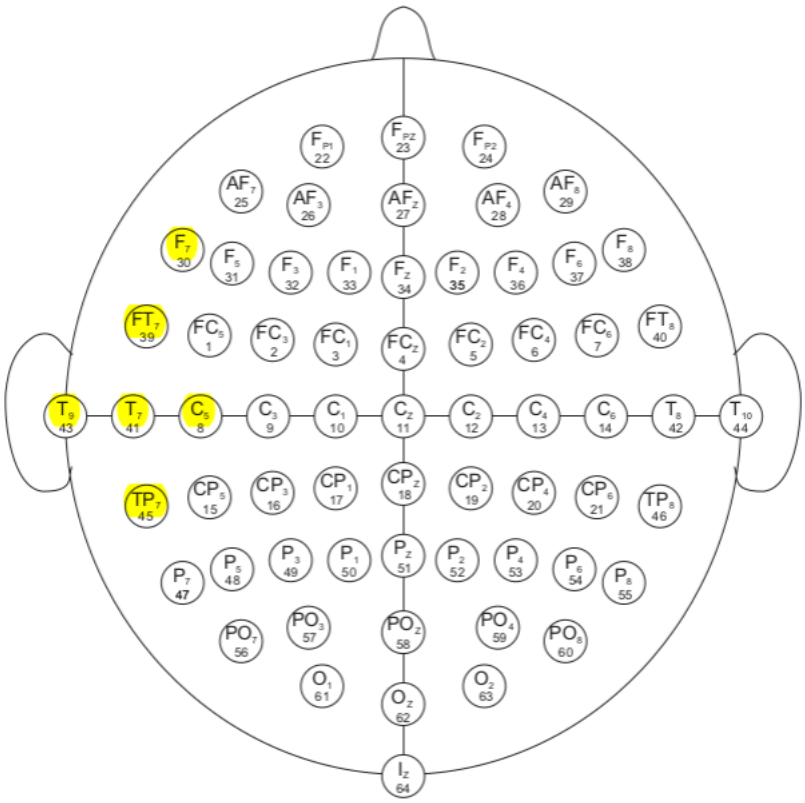



Figure 6.2: This figure shows annotations for region 1.

3. **Region 3**
   In this region we took all of the 64 channnels.

## 6.2 Logistic Regression

### 6.2.1 Logistic Function

The logistic function, also called the ***Sigmoid Function*** was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

$$\frac{1}{1+e^{-v}}$$

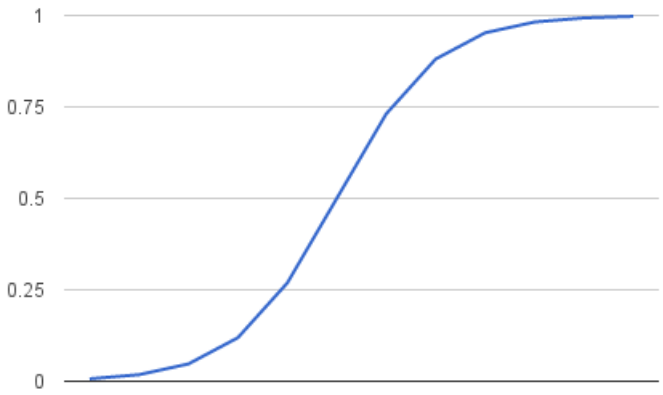



Figure 6.3: Logistic Function for $v[-5:5]$ in x-axis

logistic regression predicts the probability of an outcome that can only have two values. The prediction is based on the use of one or several predictors

$$y = \frac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}} \tag{6.1}$$

Where $y$ is the predicted output, $\beta_0$ is the bias or intercept term and $\beta_1$ is the coefficient for the single input value $(x)$

Logistic regression models the probability of the default class. if we are modeling the probability that an input (X) belongs to the default class (Y=1), we can write this formally as:

$$P(X) = P(Y = 1|X)$$

Logistic regression is a linear method, but the predictions are transformed using the logistic function. The impact of this is that we can no longer understand the predictions as a linear combination of the inputs as we can with linear regression,continuing on from above, the model can be stated as:

$$P(x) = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_1 x}} \tag{6.2}$$

we can turn around the above equation as follows:

$$\ln\left(\frac{P(x)}{1 - P(x)}\right) = \beta_0 + \beta_1 x \tag{6.3}$$

This ratio on the left is called the odds of the default class.Odds are calculated as a ratio of the probability of the event divided by the probability of not the event,e.g. $0.8/(1 - 0.8)$ which has the odds of 4. So we could instead write:

$$\ln(odds) = \beta_0 + \beta_1 x \tag{6.4}$$

Because the odds are log transformed, we call this left hand side the log-odds. We can move the exponent back to the right and write it as:

$$odds = e^{\beta_0+\beta_1 x} \tag{6.5}$$

## 6.3 Discriminant Analysis (DA)

### 6.3.1 Introduction

***Linear Discriminant Analysis(LDA)*** and ***Quadratic Discriminant Analysis(QDA)*** are two classic classifiers, with, as their names suggest, a linear and a quadratic decision surface, respectively. These classifiers are attractive because they have closed-form solutions that can be easily computed, are inherently multiclass, have proven to work well in practice, and have no hyperparameters to tune.

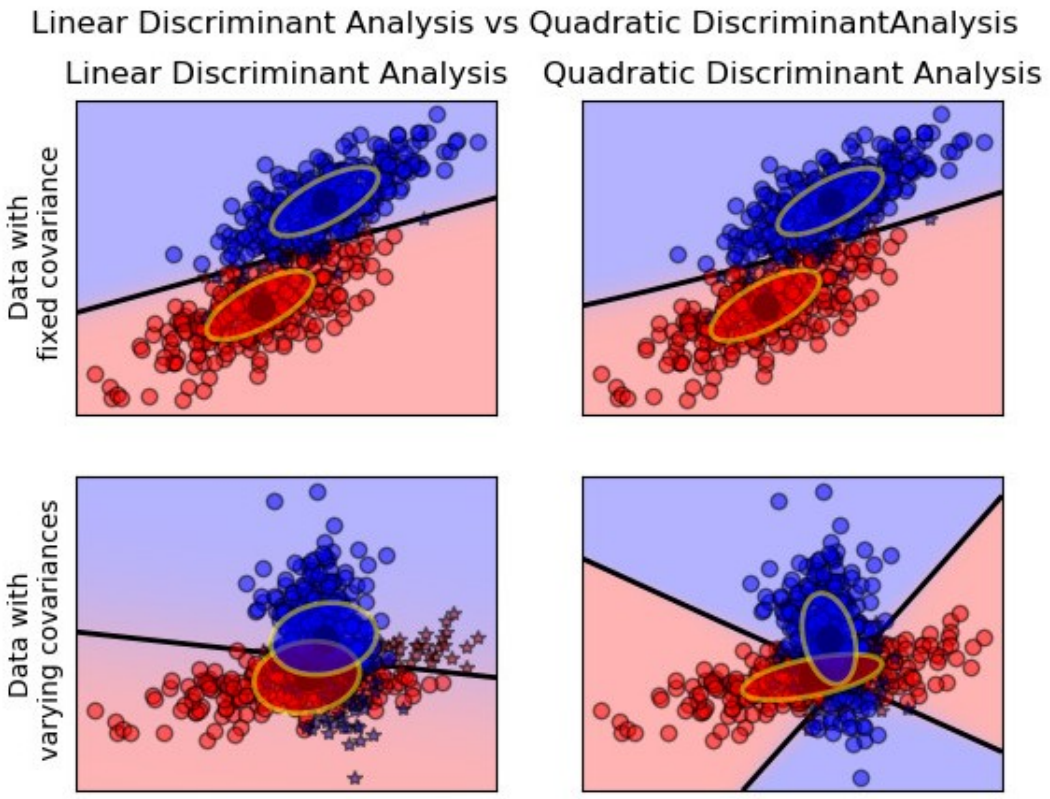



Figure 6.4: The plot shows decision boundaries for Linear Discriminant Analysis and Quadratic Discriminant Analysis. The bottom row demonstrates that Linear Discriminant Analysis can only learn linear boundaries, while Quadratic Discriminant Analysis can learn quadratic boundaries and is therefore more flexible.

### 6.3.2 Mathematical formulation of the LDA and QDA classifiers

Both LDA and QDA can be derived from simple probabilistic models which model the class conditional distribution of the data $P(X|y = k)$

$$P(y = k|X) = \frac{P(X|y = k)P(y = k)}{P(X)} = \frac{P(X|y = k)P(y = k)}{\sum_l P(X|y = l) \cdot P(y = l)} \tag{6.6}$$

and we select the class $k$ which maximizes this conditional probability.More specifically, for linear and quadratic discriminant analysis, $P(X|y)$ is modeled

as a multivariate Gaussian distribution with density:

$$P(X|y=k) = \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(X-\mu_k)^t \Sigma_k^{-1}(X-\mu_k)\right) \tag{6.7}$$

where $d$ is the number of features.To use this model as a classifier, we just need to estimate from the training data the class priors $P(X|y)$ (by the proportion of instances of class $k$), the class means $\mu_k$ (by the empirical sample class means) and the covariance matrices (either by the empirical sample class covariance matrices, or by a regularized estimator: see the section on shrinkage below).In the case of LDA, the Gaussians for each class are assumed to share the same covariance matrix: $\Sigma_k = \Sigma$ for all $k$. This leads to linear decision surfaces, which can be seen by comparing the log-probability ratios $\log[P(y=k|X)/P(y=l|X)]$:

$$\begin{aligned} \log\left(\frac{P(y=k|X)}{P(y=l|X)}\right) = \log\left(\frac{P(X|y=k)P(y=k)}{P(X|y=l)P(y=l)}\right) = 0 \Leftrightarrow \\ (\mu_k - \mu_l)^t \Sigma^{-1} X = \frac{1}{2}(\mu_k^t \Sigma^{-1}\mu_k - \mu_l^t \Sigma^{-1}\mu_l) - \log\frac{P(y=k)}{P(y=l)} \end{aligned} \tag{6.8}$$

In the case of QDA, there are no assumptions on the covariance matrices $\Sigma_k$ of the Gaussians, leading to quadratic decision surfaces.

## 6.4 K-Nearest Neighbor

The ***K-Nearest-Neighbors(KNN)*** method of classification is one of the simplest methods in machine learning. it is essentially classification by finding the most similar data points in the training data, and making an educated guess based on their classifications (KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry)
KNN falls under lazy learning, which means that there is no explicit training phase before classification it does not use the training data points to do any generalization all the training data is needed during the testing phase.The training phase is simply store all training example with its label, to make a

prediction for test case we calculate the distance to every training example and then choose the most nearest labeled point to test point and label our test point to class of that nearest point in training data.

### 6.4.1 k-Nearest Neighbor Predictions

Basic idea: Predict the label of a data point by:

1. Compute a distance between the item to be classified and every item in the training data-set.
2. Pick the k closest data points.
3. Taking a majority vote.

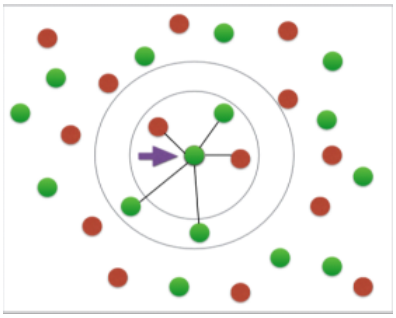


Figure 6.5: k-Nearest Neighbor Predictions

Euclidean distance is probably the one that you are most familiar.

$$d(X,Y) = ||x-y|| = \sqrt{(X-Y).(X-Y)} = \sum_{i=0}^{m}(X_i - Y_i)^2 \tag{6.9}$$

where $X_i$ and $Y_i$ are two point.
Another common metric is Cosine similarity. Rather than calculating a magnitude, Cosine similarity instead uses the difference in direction between two vectors.

$$similarity = cos(\theta) = \frac{A.B}{||A||\ ||B||} \tag{6.10}$$

### 6.4.2 Parameter selection

The best choice of $\boldsymbol{K}$ depends upon the data; generally, larger values of $\boldsymbol{K}$ reduces effect of the noise on the classification, but make boundaries between classes less distinct.
The most intuitive nearest neighbour type classifier is the one nearest neighbour classifier that assigns a point $\mathbf{X}$ to the class of its closest neighbour in the feature space, that is:

$$C_n^{1nn}(x) = Y_{(1)} \tag{6.11}$$

### 6.4.3 Distance weighted K-NN algorithm

The k-NN classification algorithm is to weigh the contribution of each of the $\boldsymbol{K}$ neighbors according to their distance to the query point $x_q$, giving greater weight $w_i$ to closer neighbors. let the closest points among the $\boldsymbol{K}$ nearest neighbors have more say in affecting the outcome of the query point. This can be achieved by introducing a set of weights $\mathbf{W}$, one for each nearest neighbor, defined by the relative closeness of each neighbor with respect to the query point:

$$W(x, p_i) = \frac{e^{-D(x,p_i)}}{\sum_{i=1}^{k} e^{-D(x,p_i)}} \tag{6.12}$$

where $D(x, p_i)$ is the distance between the query point $x$ and the $i^{th}$ case $p_i$ of the example sample.where the weight is:

$$\sum_{i=i}^{k} W(x_0, x_i) = 1 \tag{6.13}$$

### 6.4.4 kNN Computational Complexity

- Basic kNN algorithm stores all examples.
- Suppose we have $n$ examples each of dimension $d$.
- $\mathcal{O}(d)$ to compute distance to one examples.

- $\mathcal{O}(nd)$ to computed distances to all examples.
- Plus $\mathcal{O}(nk)$ time to find $k$ closest examples.
- Total time:$\mathcal{O}(nk + nd)$
- Very expensive for a large number of samples.

### 6.4.5 Reducing the complexity of KNN

- Idea: Partition space recursively and search for NN only close to the test point.
- Preprocessing: Done prior to classification process.

#### 6.4.5.1 Axis-parallel tree construction

1. Split space in direction of largest 'spread' into two equinumbered cells.
2. Repeat procedure recursively for each subcell,until some stopping criterion is achieved.

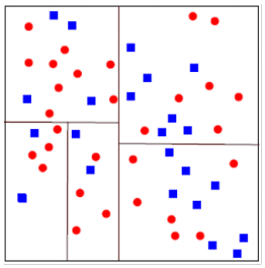


Figure 6.6: Axis-parallel tree construction

**Classification:**

1. Propagate a test point down the tree. Classification is based on NN from the final leaf reached.
2. If NN (within leaf) is further than nearest boundary - retrack

**Notes:**

- Clearly log n layers (and distance computations) suffice.
- Computation time to build tree: $\mathcal{O}(dn \log n)$
- Many variations and improvements exist (e.g. diagonal splits)
- Stopping criterion: often ad-hoc (e.g. number of points in leaf region is k, region size, etc.)

### 6.4.6 Strength and weakness of k Nearest Nieghbor

**Advantages:**

- Can be applied to the data from any distribution.
- Very simple and intuitive.
- Good classification if the number of samples is large enough.
- Robust to noisy training data (especially if we use inverse square of weighted distance as the "distance")

**Disadvantages:**

- Choosing best k may be difficult.
- Computationally heavy, but improvements possible.
- Need large number of samples for accuracy.
- Can never fix this without assuming parametric distribution.
- Computational cost is quite high because we need to compute distance of each query instance to all training samples(some indexing(k-d tree) may reduce this computational cost)

## 6.5 Naive Bayes

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable.

Bayes' theorem states the following relationship, given class variable y and dependent feature vector $x_1$ through $x_n$, :

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots x_n \mid y)}{P(x_1, \ldots, x_n)} \tag{6.14}$$

Using the naive conditional independence assumption that

$$P(x_i|y, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(x_i|y), \tag{6.15}$$

for all $i$, this relationship is simplified to

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \ldots, x_n)} \tag{6.16}$$

Since $P(x_1, \ldots, x_n)$ is constant given the input, we can use the following classification rule:

$$\begin{aligned} P(y \mid x_1, \ldots, x_n) &\propto P(y) \prod_{i=1}^{n} P(x_i \mid y) \\ &\Downarrow \\ \hat{y} &= \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y), \end{aligned} \tag{6.17}$$

and we can use Maximum A Posteriori (MAP) estimation to estimate $P(y)$ and $P(x_i \mid y)$; the former is then the relative frequency of class $y$ in the training set.

The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of $P(x_i \mid y)$.

In spite of their apparently over-simplified assumptions, naive Bayes classifiers have worked quite well in many real-world situations, famously document classification and spam filtering. They require a small amount of training data to estimate the necessary parameters. (For theoretical reasons why naive Bayes works well, and on which types of data it does, see the references below.).

Naive Bayes learners and classifiers can be extremely fast compared to more sophisticated methods. The decoupling of the class conditional feature distributions means that each distribution can be independently estimated as a one dimensional distribution. This in turn helps to alleviate problems stemming from the curse of dimensionality.

On the flip side, although naive Bayes is known as a decent classifier, it is known to be a bad estimator, so the probability outputs from predict_proba are not to be taken too seriously.

### 6.5.1 Gaussian Naive Bayes

GaussianNB implements the Gaussian Naive Bayes algorithm for classification. The likelihood of the features is assumed to be Gaussian:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \tag{6.18}$$

The parameters $\sigma_y$ and $\mu_y$ are estimated using maximum likelihood.

## 6.6 Decision Tree

A decision tree is a flowchart-like tree structure where an internal node represents feature (or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in recursively manner call recursive partitioning.

This flowchart-like structure helps you in decision making.
It's visualization like a flowchart diagram which easily mimics the human level thinking. That is why decision trees are easy to understand and interpret.

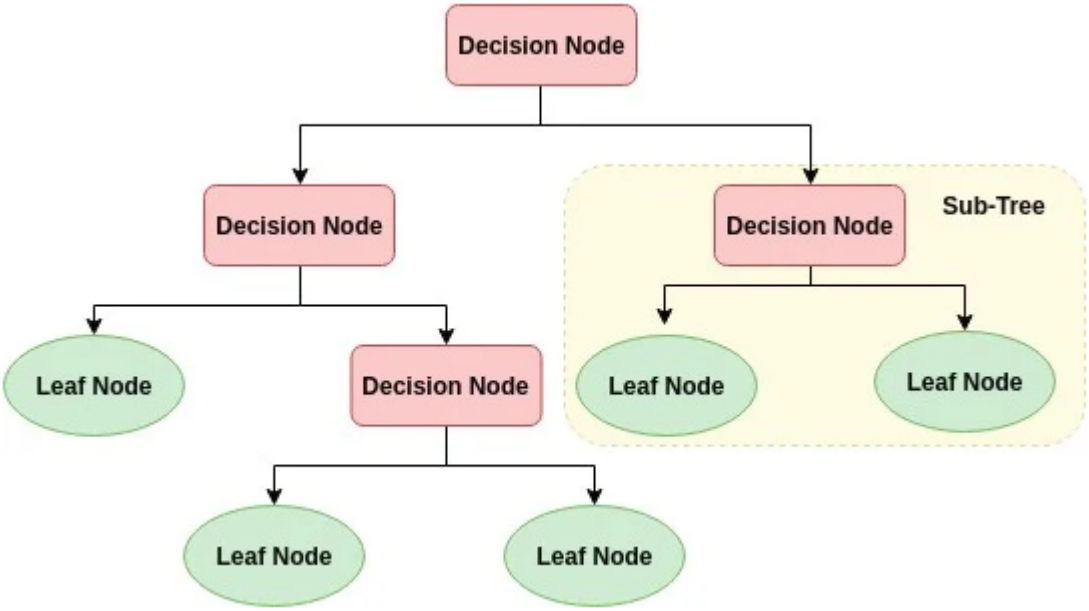



Figure 6.7: Decision Tree Algorithm

### Decision Tree algorithm

1. 1 Select the best attribute using Attribute Selection Measures(ASM) to split the records.
2. Make that attribute a decision node and breaks the dataset into smaller subsets.
3. Starts tree building by repeating this process recursively for each child until one of the condition will match:
   - All the tuples belong to the same attribute value.
   - There are no more remaining attributes.
   - There are no more instances.

**Attribute Selection Measures** is a heuristic for selecting the splitting criterion that partition data into the best possible manner . Best score attribute will be selected as a splitting attribute . Most popular selection measures are **Information Gain, Gain Ratio, and Gini Index.**

Information gain is the decrease in entropy. Information gain computes the difference between entropy before split and average entropy after split of the dataset based on given attribute values. ID3 (Iterative Dichotomiser) decision tree algorithm uses information gain.

$$Info(\mathbf{D}) = -\sum_{i=1}^{m} p_i \log_2 p_i \tag{6.19}$$

where, $p_i$ is the probability that an arbitrary tuple in $\mathbf{D}$ belongs to class $C_i$

$$Info_A(\mathbf{D}) = sum_{j=1}^{v} \frac{|\mathbf{D}_j|}{|\mathbf{D}|} \ \mathbf{X}\ Info(\mathbf{D}_j) \tag{6.20}$$

$$Gain(\mathbf{A}) = Info(\mathbf{D}) - Info_A(\mathbf{D}) \tag{6.21}$$

where $Info(\mathbf{D})$ is the average amount of information needed to identify the class label of a tuple in $\mathbf{D}$, $\frac{|\mathbf{D}_j|}{|\mathbf{D}|}$ acts as the weight of the $j^{th}$ partition, $Info_{\mathbf{A}}(\mathbf{D})$ is the expected informa-tion required to classify a tuple from $\mathbf{D}$ based on the partitioning by $\mathbf{A}$

An extension to information gain known as the gain ratio. Gain ratio handles the issue of bias by normalizing the information gain using Split Info.

$$SplitInfo(\mathbf{D}) = -\sum_{j=i}^{y} \frac{|\mathbf{D}_j|}{|\mathbf{D}|} \times \log_2 \left( \frac{|\mathbf{D}_j|}{|\mathbf{D}|} \right) \tag{6.22}$$

where $\frac{|\mathbf{D}_j|}{\mathbf{D}}$ acts as the weight of the $j^{th}$ partition $v$ is the number of discrete values in attribute $\mathbf{A}$.The gain ratio can be defined as:

$$GainRatio(\mathbf{A}) = \frac{Gain(\mathbf{A})}{SplitInfo_A(\mathbf{A}))} \tag{6.23}$$

## 6.7 Support Vector Machine(SVM)

### 6.7.1 Intoduction

A ***Support Vector Machine(SVM)*** is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimentional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.

### 6.7.2 Linear SVM

We are given a training dataset of $n$ points of the form $(\vec{x}_1, y_1), ..., (\vec{x}_n, y_n)$ where the $y_i$ are either 1 or $-1$, each indicating the class to which the point $\vec{x}_i$ belongs. Each $\vec{x}_i$ is a $p$-dimensional real vector. We want to find the "maximum-margin hyperplane" that divides the group of points $\vec{x}_i$ for which $y_i = 1$ from the group of points for which $y_i = -1$, which is defined so that the distance between the hyperplane and the nearest point $\vec{x}_i$ from either group is maximized.
Any hyperplane can be written as the set of points $\vec{x}$ satisfying

$$\vec{w} \cdot \vec{x} - b = 0 \tag{6.24}$$

where $\vec{w}$ is the normal vector to the hyperplane. except that $\vec{w}$ is not necessarily a unit vector. The parameter $\frac{b}{\|\vec{w}\|}$ determines the offset of the hyperplane from the origin along the normal vector $\vec{w}$.

### 6.7.3 Kernal

The learning of the hyperplane in linear SVM is done by transforming the problem using some linear algebra. This is where the kernel plays role.

For **linear kernel** the equation for prediction for a new input using the dot product between the input $(x)$ and each support vector $(x_i)$ is calculated as follows:

$$\langle x \cdot x_i \rangle \tag{6.25}$$

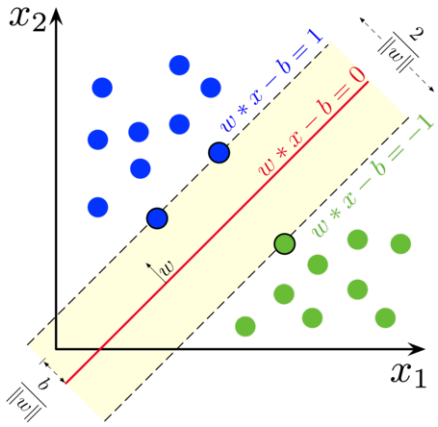



Figure 6.8: Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called the support vectors.

This is an equation that involves calculating the inner products of a new input vector $(x)$ with all support vectors in training data

For **polynomial kernel** can be written as

$$(\gamma\langle x, x'\rangle + r)^d \tag{6.26}$$

For **radial basis function** can be written as

$$\exp(-\gamma\|x - x'\|^2) \tag{6.27}$$

For **sigmoid**

$$\tanh(\gamma\langle x, x'\rangle + r) \tag{6.28}$$

### 6.7.4 Solve SVM

Given training vectors $x_i \in \mathbb{R}^p, i = 1, \ldots, n$, in two classes, and a vector, $y \in \{1, -1\}^n$ SVM solves the following primal problem:

$$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i \tag{6.29}$$

$$\text{subject to } y_i(w^T \phi(x_i) + b) \geq 1 - \zeta_i, \zeta_i \geq 0, i = 1, ..., n$$

it's dual is :

$$\min_{\alpha} \frac{1}{2}\alpha^T Q\alpha - e^T\alpha \tag{6.30}$$
$$\text{subject to } y^T\alpha = 0 \ , \ 0 \leq \alpha_i \leq C \ , \ i = 1, ..., n$$

where $e$ is the vector of all ones, $C > 0$ is the upper bound, $Q$ is an $n$ by $n$ positive semidefinite matrix, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$, where $K(x_i, x_j) = \phi(x_i)^T\phi(x_j)$ is the kernel. Here training vectors are implicitly mapped into a higher (maybe infinite) dimensional space by the function $\phi$.

The decision function is:

$$\text{sgn}(\sum_{i=1}^{n} y_i\alpha_i K(x_i, x) + \rho) \tag{6.31}$$

## 6.8 Classification Matric

When evaluating a classifier, there are different ways of measuring its performance. For supervised learning with two possible classes, all measures of performance are based on four numbers obtained from applying the classifier to the test set. These numbers are called true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN)

| | | Actual | |
|---|---|---|---|
| | | Positive | Negative |
| Predictive | Positive | TP | FP |
| | Negative | FN | TN |

A table like the one above is called a confusion matrix. The terminology true positive, etc., is standard, but whether columns correspond to predicted and rows to actual, or vice versa, is not standard.

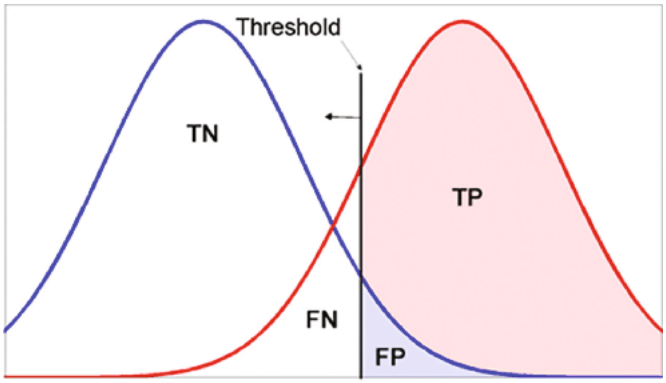



Figure 6.9: Probability distributions for two classes.

## 6.9 Receiver Operating Characteristic (ROC)

### 6.9.1 Introduction

A receiver operating characteristic curve, or ROC curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the ***true positive rate (TPR)*** against the ***false positive rate (FPR)*** at various threshold settings. The true-positive rate is also known as sensitivity, recall or probability of detection in machine learning. The false-positive rate is also known as the fall-out or probability of false alarm and can be calculated as (1 - specificity). It can also be thought of as a plot of the power as a function of the Type I Error of the decision rule (when the performance is calculated from just a sample of the population, it can be thought of as estimators of these quantities). The ROC curve is thus the sensitivity as a function of fall-out. In general, if the probability distributions for both detection and false alarm are known, the ROC curve can be generated by plotting the cumulative distribution function (area under the probability distribution from $-\infty$ to the discrimination threshold) of the detection probability in the y-axis versus the cumulative distribution function of the false-alarm probability on the x-axis. ROC analysis provides tools to select possibly optimal models and to discard suboptimal ones independently from (and prior to specifying) the cost context or the class distribution. ROC analysis is related

in a direct and natural way to cost/benefit analysis of diagnostic decision making.

The ROC curve was first developed by electrical engineers and radar engineers during World War II for detecting enemy objects in battlefields and was soon introduced to psychology to account for perceptual detection of stimuli. ROC analysis since then has been used in medicine, radiology, biometrics, forecasting of natural hazards, meteorology, model performance assessment, and other areas for many decades and is increasingly used in machine learning and data mining research.

The ROC is also known as a relative operating characteristic curve, because it is a comparison of two operating characteristics (TPR and FPR) as the criterion changes.

### 6.9.2 Basis Concept

A classification model (classifier or diagnosis) is a mapping of instances between certain classes/groups. The classifier or diagnosis result can be a real value (continuous output), in which case the classifier boundary between classes must be determined by a threshold value (for instance, to determine whether a person has hypertension based on a blood pressure measure). Or it can be a discrete class label, indicating one of the classes.

Consider a two-class prediction problem (binary classification), in which the outcomes are labeled either as positive (p) or negative (n). There are four possible outcomes from a binary classifier. If the outcome from a prediction is p and the actual value is also p, then it is called a true positive (TP); however if the actual value is n then it is said to be a false positive (FP). Conversely, a true negative (TN) has occurred when both the prediction outcome and the actual value are n, and false negative (FN) is when the prediction outcome is n while the actual value is p.

To get an appropriate example in a real-world problem, consider a diagnostic test that seeks to determine whether a person has a certain disease. A false positive in this case occurs when the person tests positive, but does not actually have the disease. A false negative, on the other hand, occurs when the person tests negative, suggesting they are healthy, when they actually do

have the disease.

Let us define an experiment from P positive instances and N negative instances for some condition. The four outcomes can be formulated in a 2x2 contingency table or confusion matrix, as follows:

| | | True condition | | | |
|---|---|---|---|---|---|
| | Total population | Condition positive | Condition negative | Prevalence = $\frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$ | Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$ |
| **Predicted condition** | Predicted condition positive | **True positive** | **False positive**, Type I error | Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$ | False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$ |
| | Predicted condition negative | **False negative**, Type II error | **True negative** | False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$ | Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$ |
| | | True positive rate (TPR), Recall, Sensitivity, probability of detection, Power = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$ | False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$ | Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$ | Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR−}}$ | $F_1$ score = $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$ |
| | | False negative rate (FNR), Miss rate = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$ | Specificity (SPC), Selectivity, True negative rate (TNR) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$ | Negative likelihood ratio (LR−) = $\frac{\text{FNR}}{\text{TNR}}$ | | |

Figure 6.10: The four outcomes formulated in a 2x2 contingency table or confusion matrix.

### 6.9.3 ROC Space

The contingency table can derive several evaluation "metrics" (see infobox). To draw an ROC curve, only the true positive rate (TPR) and false positive rate (FPR) are needed (as functions of some classifier parameter). The TPR defines how many correct positive results occur among all positive samples available during the test. FPR, on the other hand, defines how many incorrect positive results occur among all negative samples available during the test.

An ROC space is defined by FPR and TPR as x and y axes, respectively, which depicts relative trade-offs between true positive (benefits) and false positive (costs). Since TPR is equivalent to sensitivity and FPR is equal

to 1 - specificity, the ROC graph is sometimes called the sensitivity vs (1 - specificity) plot. Each prediction result or instance of a confusion matrix represents one point in the ROC space.

The best possible prediction method would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing 100% sensitivity (no false negatives) and 100

The diagonal divides the ROC space. Points above the diagonal represent good classification results (better than random); points below the line represent bad results (worse than random). Note that the output of a consistently bad predictor could simply be inverted to obtain a good predictor.

Let us look into four prediction results from 100 positive and 100 negative instances:

| A | | | B | | | C | | | C' | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| TP=63 | FP=28 | 91 | TP=77 | FP=77 | 154 | TP=24 | FP=88 | 112 | TP=76 | FP=12 | 88 |
| FN=37 | TN=72 | 109 | FN=23 | TN=23 | 46 | FN=76 | TN=12 | 88 | FN=24 | TN=88 | 112 |
| 100 | 100 | 200 | 100 | 100 | 200 | 100 | 100 | 200 | 100 | 100 | 200 |
| TPR = 0.63 | | | TPR = 0.77 | | | TPR = 0.24 | | | TPR = 0.76 | | |
| FPR = 0.28 | | | FPR = 0.77 | | | FPR = 0.88 | | | FPR = 0.12 | | |
| PPV = 0.69 | | | PPV = 0.50 | | | PPV = 0.21 | | | PPV = 0.86 | | |
| F1 = 0.66 | | | F1 = 0.61 | | | F1 = 0.23 | | | F1 = 0.81 | | |
| ACC = 0.68 | | | ACC = 0.50 | | | ACC = 0.18 | | | ACC = 0.82 | | |

Figure 6.11: Four prediction results from 100 positive and 100 negative instances.

Plots of the four results above in the ROC space are given in the figure. The result of method A clearly shows the best predictive power among A, B, and C. The result of B lies on the random guess line (the diagonal line), and it can be seen in the table that the accuracy of B is 50 %. However, when C is mirrored across the center point (0.5,0.5), the resulting method C is even better than A. This mirrored method simply reverses the predictions of whatever method or test produced the C contingency table. Although the original C method has negative predictive power, simply reversing its decisions leads to a new predictive method C which has positive predictive

power. When the C method predicts p or n, the C method would predict n or p, respectively. In this manner, the C test would perform the best. The closer a result from a contingency table is to the upper left corner, the better it predicts, but the distance from the random guess line in either direction is the best indicator of how much predictive power a method has. If the result is below the line (i.e. the method is worse than a random guess), all of the method's predictions must be reversed in order to utilize its power, thereby moving the result above the random guess line.

### 6.9.4 Curves in ROC space

In binary classification, the class prediction for each instance is often made based on a continuous random variable $X$, which is a "score" computed for the instance (e.g. the estimated probability in logistic regression). Given a threshold parameter $T$, the instance is classified as "positive" if $X > T$, and "negative" otherwise. $X$ follows a probability density $f_1(x)$ if the instance actually belongs to class "positive", and $f_0$ if otherwise. Therefore, the true positive rate is given by $\mathrm{TPR}(T) = \int_T^\infty f_1(x)dx$ and the false positive rate is given by $\mathrm{FPR}(T) = \int_T^\infty f_0(x)dx$. The ROC curve plots parametrically TPR(T) versus FPR(T) with T as the varying parameter.

For example, imagine that the blood protein levels in diseased people and healthy people are normally distributed with means of 2 g/dL and 1 g/dL respectively. A medical test might measure the level of a certain protein in a blood sample and classify any number above a certain threshold as indicating disease. The experimenter can adjust the threshold (black vertical line in the figure), which will in turn change the false positive rate. Increasing the threshold would result in fewer false positives (and more false negatives), corresponding to a leftward movement on the curve. The actual shape of the curve is determined by how much overlap the two distributions have.

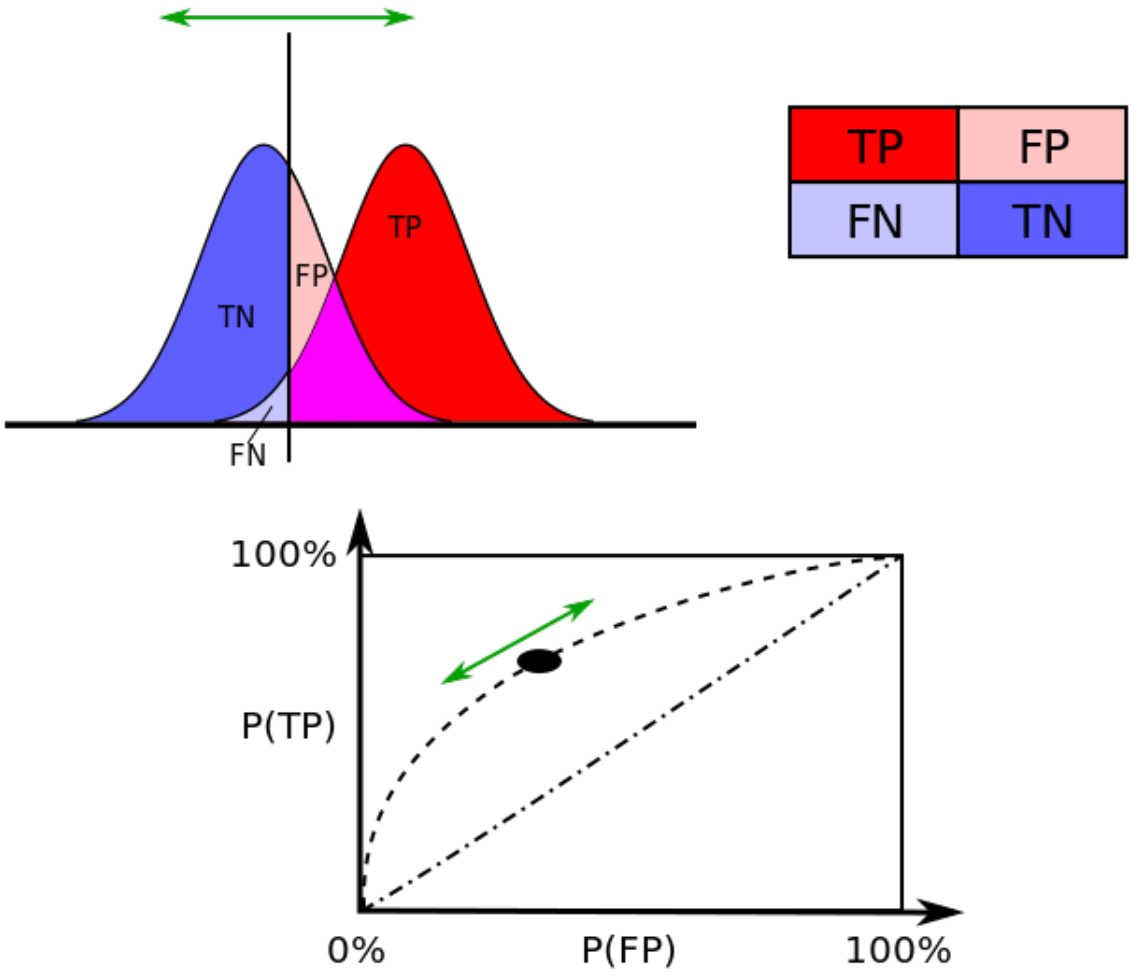



Figure 6.12: This figure shows Receiver Operating Characteristic (ROC) curves

# Chapter 7

# Software Design Description

# 7.1 Introduction

In this chapter we will talk about ***Software Design Description*** which is a written description of a software product, that a software designer writes in order to give a software development team overall guidance to the architecture of the software project. First we will talk about the ***Block Diagram***, then we will start talking about our ***Activity Diagram***, and finally we will talk about the ***Class Diagram*** of the software.

# 7.2 Block Diagram

#### 7.2.0.1 Introduction

A block diagram is a diagram of a system in which the principal parts or functions are represented by blocks connected by lines that show the relationships of the blocks. They are heavily used in engineering in hardware design, electronic design, software design, and process flow diagrams. Block diagrams are typically used for higher level, less detailed descriptions that are intended to clarify overall concepts without concern for the details of implementation. Contrast this with the schematic diagrams and layout diagrams used in electrical engineering, which show the implementation details of electrical components and physical construction. lock diagrams rely on the principle of the black box where the contents are hidden from view either to avoid being distracted by the details or because the details are not known. We know what goes in, we know what goes out, but we can't see how the box does its work.

#### 7.2.0.2 Our Block Diagram

# 7.3 Activity Diagram

#### 7.3.0.1 Introduction

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to

represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

#### 7.3.0.2 Our Activity Diagram

## 7.4 Class Diagram

#### 7.4.0.1 Introduction

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram. The purpose of class diagram is to model the static view of an application. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction.

#### 7.4.0.2 Our Class Diagram

# Chapter 8

# Results And Experiment

## 8.1 Introduction

After a lot of experimenting and evaluation we choosed the best combination of the parameters for each classifier.

## 8.2 Offline Experiment

This data set was created and contributed to PhysioBank by Gerwin Schalk (schalk at wadsworth dot org) and his colleagues at the BCI Program, Wadsworth Center, New York State Department of Health, Albany, NY. W.A. Sarnacki collected the data. Aditya Joshi compiled the dataset and prepared the documentation.
The data used in offline mode was collected from subjects performed different motor imagery tasks while 64-channel EEG were recorded using the BCI2000 system . Each subject performed 14 experimental runs: two one-minute baseline runs (one with eyes open, one with eyes closed), and three two-minute runs of each of the four following tasks:

1. A target appears on either the left or the right side of the screen. The subject opens and closes the corresponding fist until the target disappears. Then the subject relaxes.

2. A target appears on either the left or the right side of the screen. The subject imagines opening and closing the corresponding fist until the target disappears. Then the subject relaxes.

3. A target appears on either the top or the bottom of the screen. The subject opens and closes either both fists (if the target is on top) or both feet (if the target is on e bottom) until the target disappears. Then the subject relaxes.

4. A target appears on either the top or the bottom of the screen. The subject imagines opening and closing either both fists (if the target is on top) or both feet (if the target is on the bottom) until the target disappears. Then the subject relaxes.

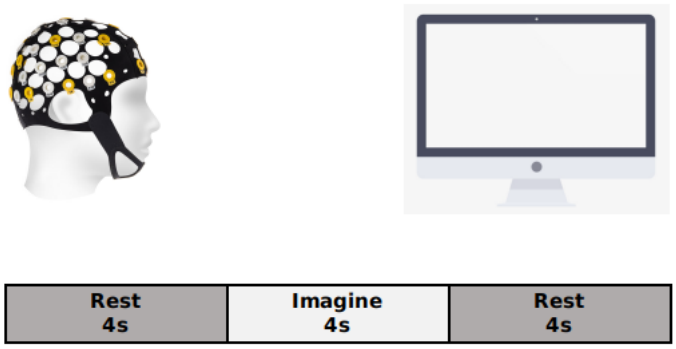



Figure 8.1: A target appears on either the left or the right side of the screen. The subject imagines opening and closing the corresponding fist until the target disappears. Then the subject relaxes

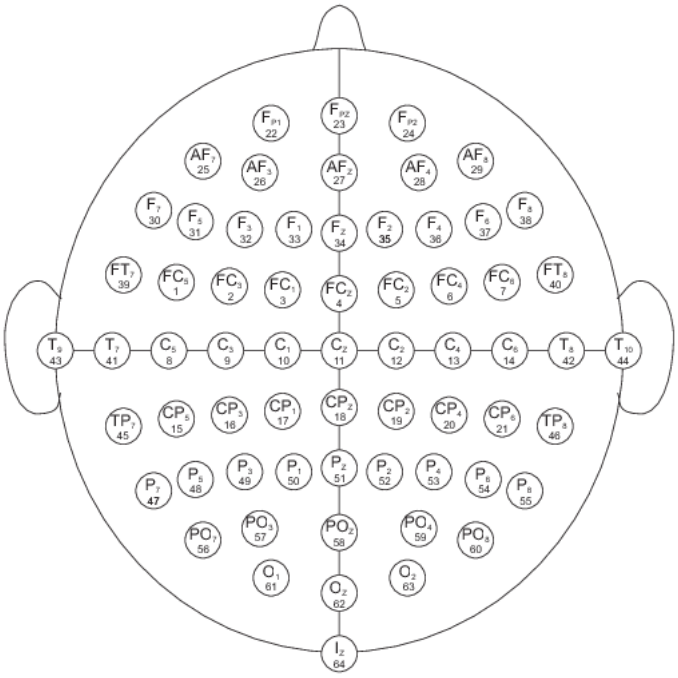


Figure 8.2: the experimental runs were imagine opening and closing left or right fist , The data are provided here in EDF+ format (containing 64 EEG signals, each sampled at 160 samples per second, and an annotation channel).

## 8.2.1 Experiments

### 8.2.1.1 Dependent Subjects

**First Experiment** In this Experiment we used Linear Discriminant Analysis(LDA) with solver least square solution and shrinkage with ledoit-wolf lemma algorithm. Number of component of PCA in this Experiment was 13 and transform data using fourier transform and take the first 10 frequencies (Region 1 and 64 channels). In Region 2 we used transform data using fourier transform and take the first 10 frequencies without PCA. The results of this experiment showed in figure 8.3.

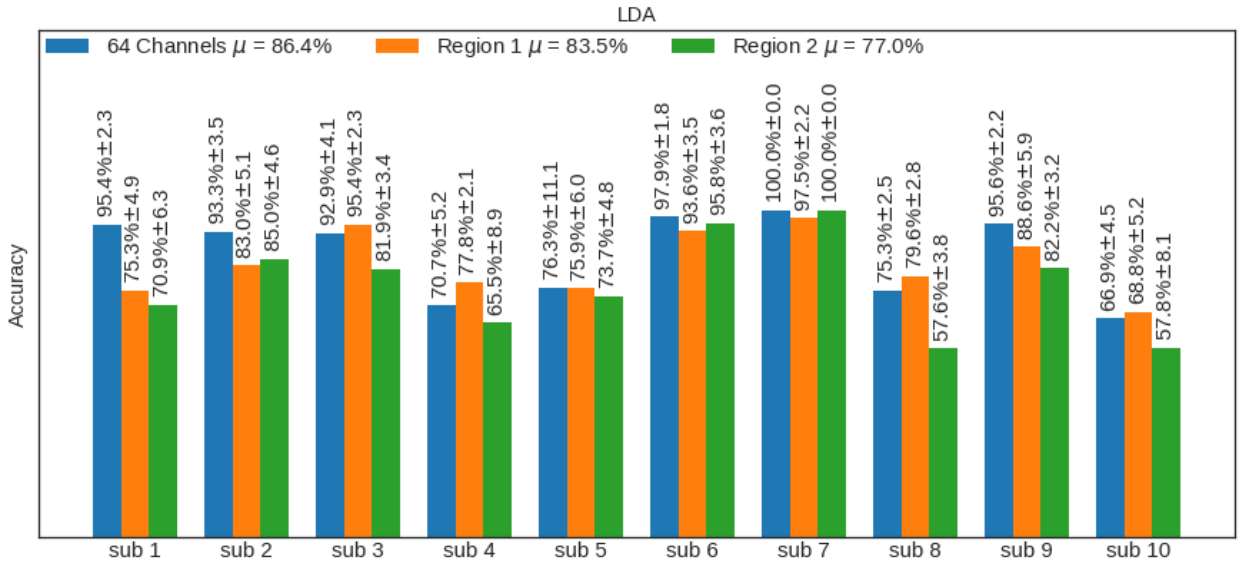



Figure 8.3: This figure shows the mean accuracy and the standard deviation(STD) over 10 subjects.

**Second Experiment** In this Experiment we used Naive Bayes(Gaussian Naive Bayes) The number of component of PCA was 13 and transfer data with Fourier Transform and take first 10 frequencies (Region 1 and 64 channels). In Region 2 we used transform data using fourier transform and take the first 10 frequencies without PCA. The results of this experiment showed in figure 8.4.

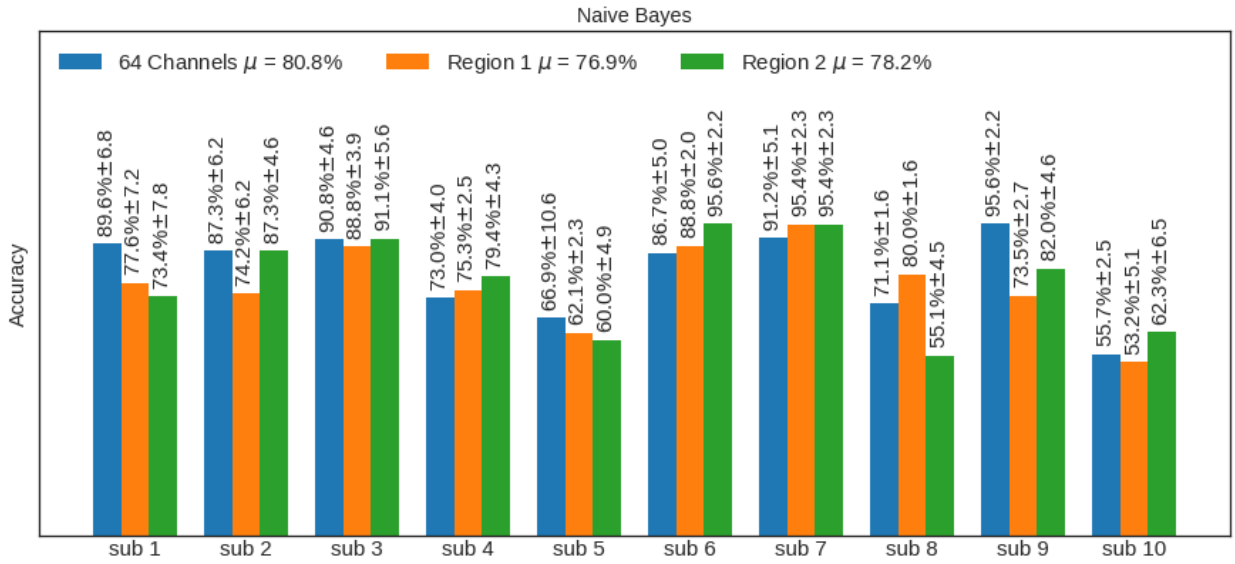



Figure 8.4: This figure shows the mean accuracy and the standard deviation(STD) over 10 subjects.

**Third Experiment** In this Experiment we used Support Vector Machine (SVM) with kernal linear and C=2.0 . The number of component of ICA was 6 and transform data to Fourier Transform and take the first 10 frequencies (Region 1, Region 2 and 64 channels). The results of this experiment showed in figure 8.5.

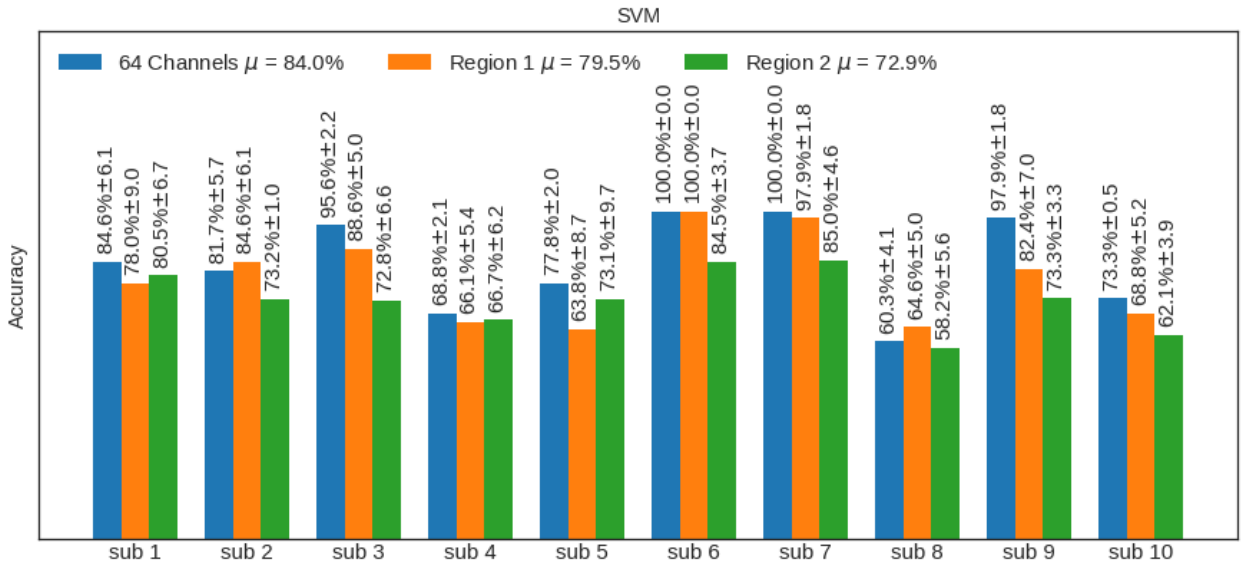



Figure 8.5: This figure shows the mean accuracy and the standard deviation(STD) over 10 subjects.

**Forth Expriement** In this Experiment we used k-Nearest Neighbors(KNN) with 3 nearest neighbors. The number of component of ICA was 5, and transform data with Fourier Transform and take the first 10 frequencies (Region 1, Region 2 and 64 channels) . The results of this experiment showed in figure 8.6.

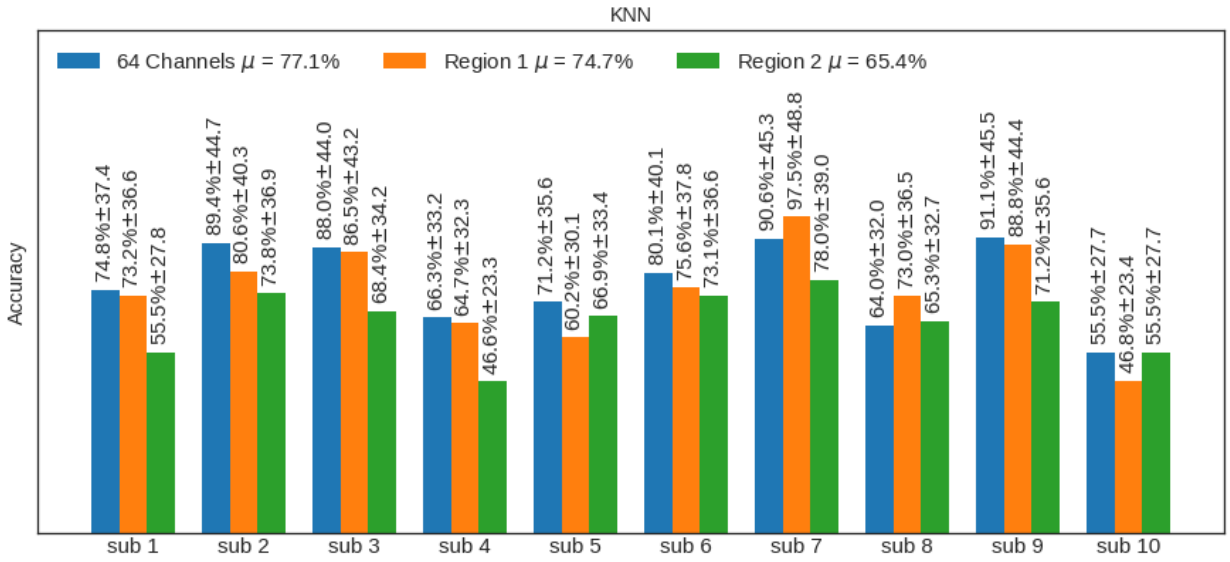



Figure 8.6: This figure shows the mean accuracy and the standard deviation(STD) over 10 subjects.

**Fifth Experiment** In this Experiment we used Logistic Regression with solver newton-cg and C=600 , The number of component of ICA was 9, and transform data with Fourier Transform and take the first 10 frequencies (Region 1 and 64 channels).In Region 2 we used transform data using fourier transform and take the first 10 frequencies without ICA. The results of this experiment showed in figure 8.7.

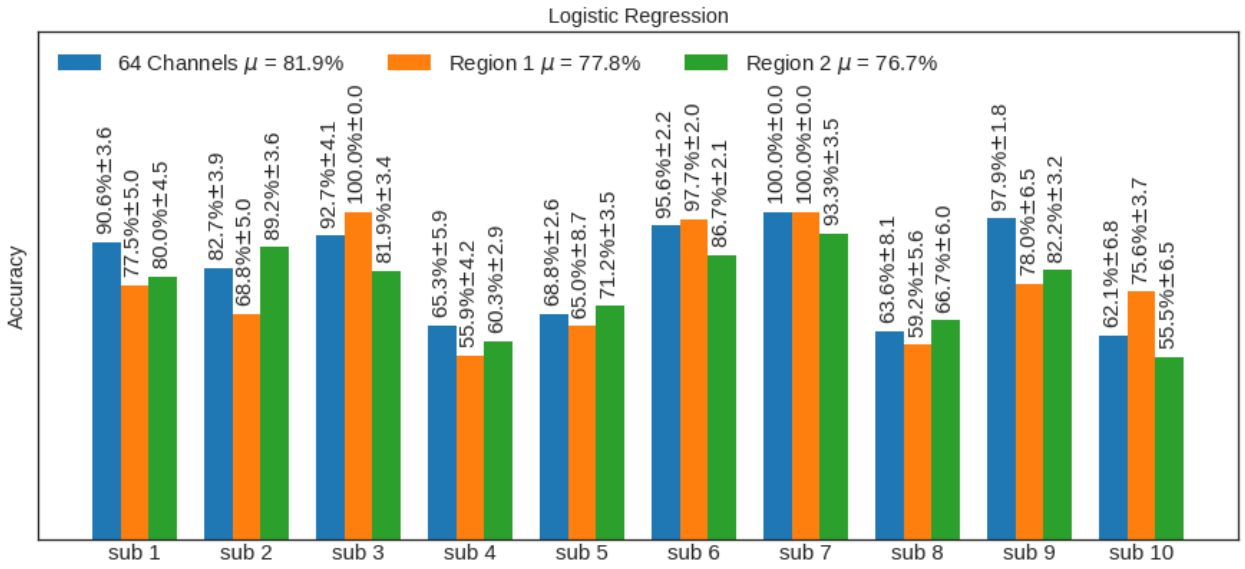



Figure 8.7: This figure shows the mean accuracy and the standard deviation(STD) over 10 subjects.

**Sixth Experiment** In this Experiment we used Decision Tree, The number of component of ICA was 9, and transform data with Fourier Transform and take the first 10 frequencies. The results of this experiment showed in figure 8.8.

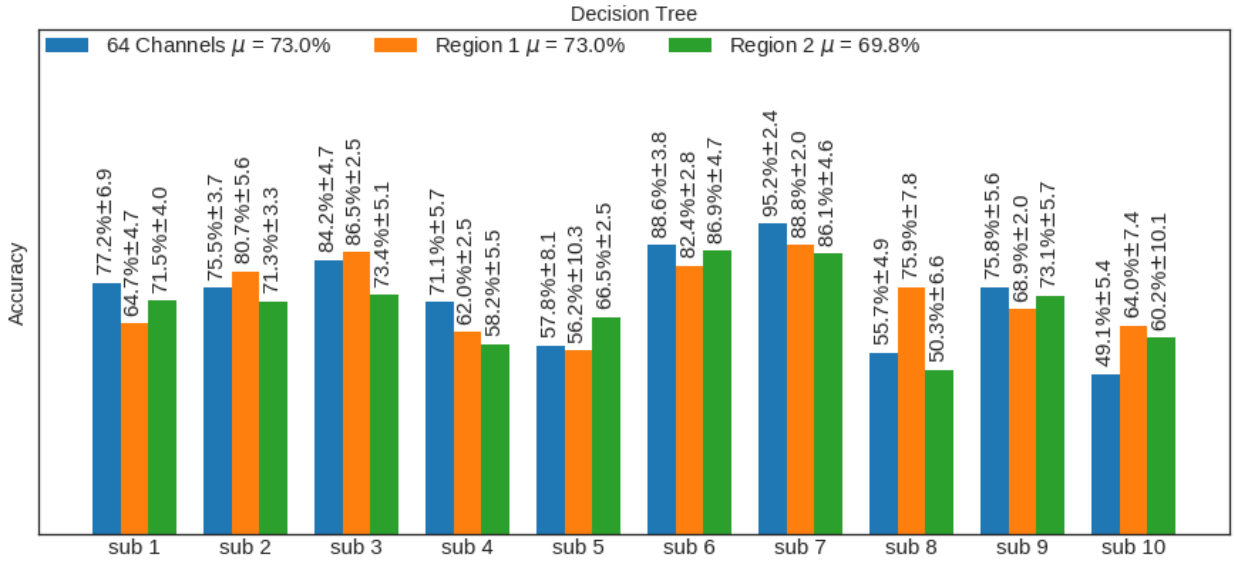



Figure 8.8: This figure shows the mean accuracy and the standard deviation(STD) over 10 subjects.

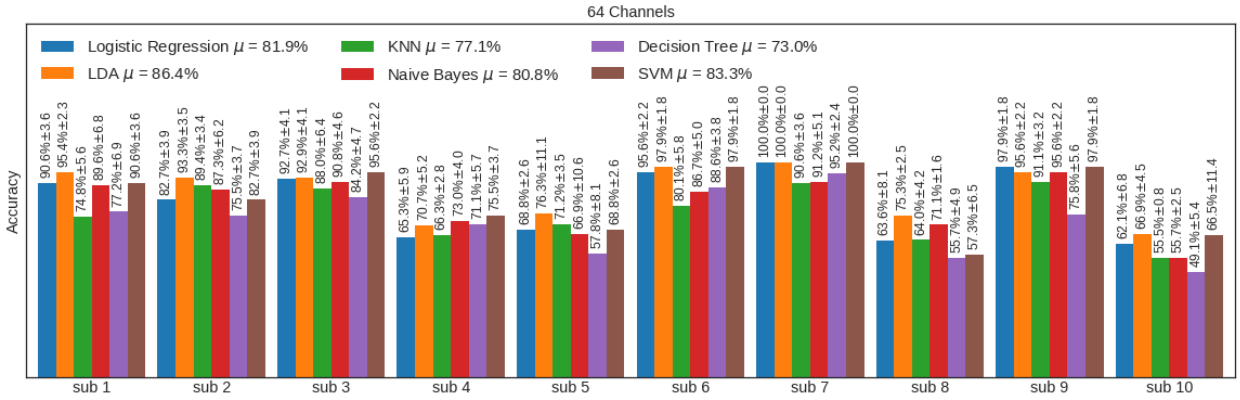



Figure 8.9: This figure shows the mean accuracy and the standard deviation(STD) over 10 subjects for each classifier(64 channels).

| Feature Extraction | PCA | | ICA | | | |
|---|---|---|---|---|---|---|
| Classification | LDA | NB | SVM | KNN | LOG | DT |
| Sub 1 | 95.4% ± 2.3 | 89.6% ± 6.8 | 90.6% ± 3.6 | 74.8% ± 5.6 | 90.6% ± 3.6 | 77.2% ± 6.9 |
| Sub 2 | 93.3% ± 3.5 | 87.3% ± 6.2 | 82.7% ± 3.9 | 89.4% ± 3.4 | 82.7% ± 3.9 | 75.5% ± 3.7 |
| Sub 3 | 92.9% ± 4.1 | 90.8% ± 4.6 | 95.6% ± 2.2 | 88% ± 6.4 | 92.7% ± 4.1 | 84.2% ± 4.7 |
| Sub 4 | 70.7% ± 5.2 | 73% ± 4 | 75.5% ± 3.7 | 66.3% ± 2.8 | 65.3% ± 5.9 | 71.1% ± 5.7 |
| Sub 5 | 76.3% ± 11.1 | 66.9% ± 10.6 | 68.8% ± 2.6 | 71.2% ± 3.5 | 68.8% ± 2.6 | 57.8% ± 8.1 |
| Sub 6 | 97.9% ± 1.8 | 86.7% ± 5 | 97.9% ± 1.8 | 80.1% ± 5.8 | 95.6% ± 2.2 | 88.6% ± 3.8 |
| Sub 7 | 100% ± 0 | 91.2% ± 5.1 | 100% ± 0 | 90.6% ± 3.6 | 100% ± 0 | 95.2% ± 2.4 |
| Sub 8 | 75.3% ± 2.5 | 71.1% ± 1.6 | 57.3% ± 6.5 | 64% ± 4.2 | 63.6% ± 8.1 | 55.7% ± 4.9 |
| Sub 9 | 95.6% ± 2.2 | 95.6% ± 2.2 | 97.9% ± 1.8 | 71.1% ± 3.2 | 97.9% ± 1.8 | 75.8% ± 5.6 |
| Sub 10 | 66.9% ± 4.5 | 55.7% ± 2.5 | 66.5% ± 11.4 | 55.5% ± 0.8 | 62.1% ± 6.8 | 49.1% ± 5.4 |
| Mean | 86.4% | 80.8% | 83.3% | 77.1% | 81.9% | 73% |

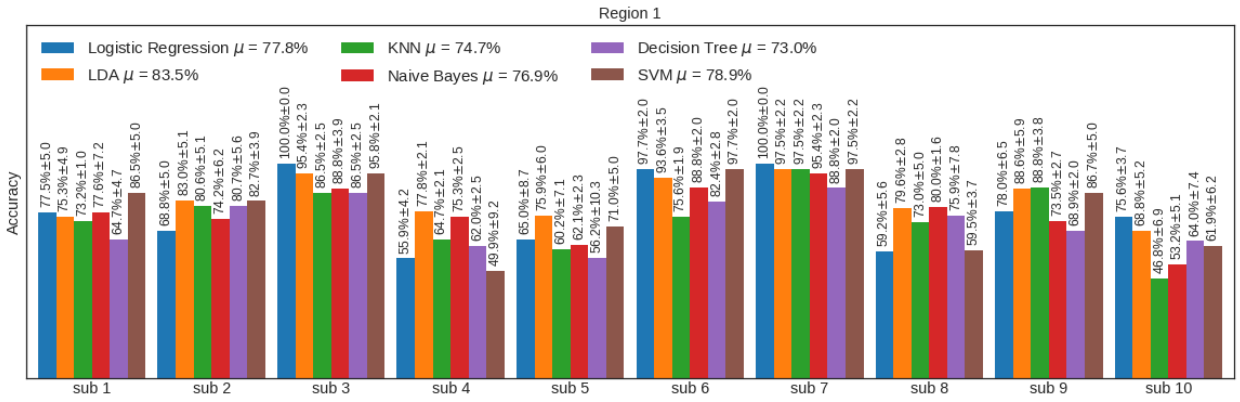



Figure 8.10: This figure shows the mean accuracy and the standard deviation(STD) over the 10 subjects on region 1 for each classifier.

| Feature Extraction | PCA | | ICA | | | |
|---|---|---|---|---|---|---|
| Classification | LDA | NB | SVM | KNN | LOG | DT |
| Sub 1 | 75.3% ± 4.9 | 77.6% ± 7.2 | 86.5% ± 5 | 73.2% ± 1 | 77.5% ± 5 | 64.7% ± 4.7 |
| Sub 2 | 83% ± 5.1 | 74.2% ± 6.2 | 82.7% ± 3.9 | 80.6% ± 5.1 | 68.8% ± 5 | 80.7% ± 5.6 |
| Sub 3 | 95.4% ± 2.3 | 88.8% ± 3.9 | 95.8% ± 2.1 | 86.5% ± 2.5 | 100% ± 0 | 86.5% ± 2.5 |
| Sub 4 | 77.8% ± 2.1 | 75.3% ± 2.5 | 49.9% ± 9.2 | 64.7% ± 2.1 | 55.9% ± 4.2 | 62% ± 2.5 |
| Sub 5 | 75.9% ± 6 | 62.1% ± 2.3 | 71% ± 5 | 60.2% ± 7.1 | 65% ± 8.7 | 56.2% ± 10.3 |
| Sub 6 | 93.6% ± 3.5 | 88.8% ± 2 | 97.7% ± 2 | 75.6% ± 1.9 | 97.7% ± 2 | 82.4% ± 2.8 |
| Sub 7 | 97.5% ± 2.2 | 95.4% ± 2.3 | 97.5% ± 2.2 | 97.5% ± 2.2 | 100% ± 0 | 88.8% ± 2 |
| Sub 8 | 79.5% ± 2.8 | 80% ± 1.6 | 59.5% ± 3.7 | 73% ± 5 | 59.2% ± 5.6 | 75.9% ± 7.8 |
| Sub 9 | 88.6% ± 5.9 | 73.5% ± 2.7 | 86.7% ± 5 | 88.8% ± 3.8 | 78% ± 6.5 | 68.9% ± 2 |
| Sub 10 | 68.8% ± 5.2 | 53.2% ± 5.1 | 61.9% ± 6.2 | 46.8% ± 6.9 | 75.6% ± 3.7 | 64% ± 7.4 |
| Mean | 83.5% | 76.9% | 78.9% | % | 77.8% | 73% |

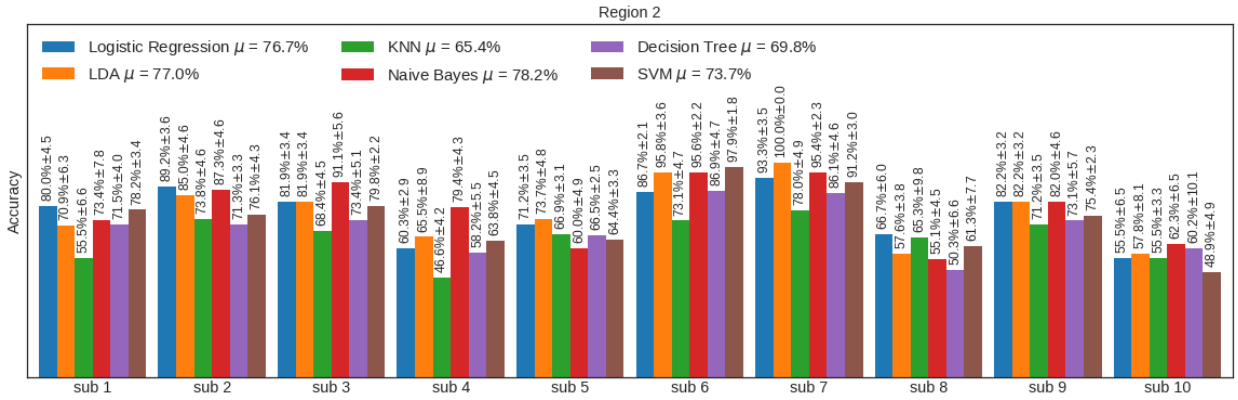



Figure 8.11: This figure shows the mean accuracy and the standard deviation(STD) over the 10 subjects on region 2 for each classifier.

| Classification | LDA | NB | SVM | KNN | LOG | DT |
|---|---|---|---|---|---|---|
| Sub 1 | $70.9\% \pm 6.3$ | $73.4\% \pm 7.8$ | $78.2\% \pm 3.4$ | $55.5\% \pm 6.6$ | $80\% \pm 4.5$ | $71.5\% \pm 4$ |
| Sub 2 | $85\% \pm 4.6$ | $87.3\% \pm 4.6$ | $76.1\% \pm 4.3$ | $73.8\% \pm 4.6$ | $89.2\% \pm 3.6$ | $71.3\% \pm 3.3$ |
| Sub 3 | $81.9\% \pm 3.4$ | $91.1\% \pm 5.6$ | $79.8\% \pm 2.2$ | $68.4\% \pm 4.5$ | $81.9\% \pm 3.4$ | $73.4\% \pm 5.1$ |
| Sub 4 | $65.5\% \pm 8.9$ | $79.4\% \pm 4.3$ | $63.8\% \pm 4.5$ | $46.6\% \pm 4.2$ | $60.3\% \pm 2.9$ | $58.2\% \pm 5.5$ |
| Sub 5 | $73.7\% \pm 4.8$ | $60\% \pm 4.9$ | $64.4\% \pm 3.3$ | $66.9\% \pm 3.1$ | $71.2\% \pm 3.5$ | $66.5\% \pm 2.5$ |
| Sub 6 | $95.8\% \pm 3.6$ | $95.6\% \pm 2.2$ | $97.9\% \pm 1.8$ | $73.1\% \pm 4.7$ | $86.7\% \pm 2.1$ | $86.9\% \pm 4.7$ |
| Sub 7 | $100\% \pm 0$ | $95.4\% \pm 2.3$ | $91.2\% \pm 3$ | $78\% \pm 4.9$ | $93.3\% \pm 3.5$ | $86.1\% \pm 4.6$ |
| Sub 8 | $57.6\% \pm 3.8$ | $55.1\% \pm 4.5$ | $61.3\% \pm 7.7$ | $65.3\% \pm 9.8$ | $66.7\% \pm 6$ | $50.3\% \pm 6.6$ |
| Sub 9 | $82.2\% \pm 3.2$ | $82\% \pm 4.6$ | $75.4\% \pm 2.3$ | $71.2\% \pm 3.5$ | $82.2\% \pm 3.2$ | $73.1\% \pm 5.7$ |
| Sub 10 | $57.8\% \pm 8.1$ | $62.3\% \pm 6.5$ | $48.9\% \pm 4.9$ | $55.5\% \pm 3.3$ | $55.5\% \pm 6.5$ | $60.2\% \pm 10.1$ |
| Mean | 77% | 78.2% | 73.7% | 65.4% | 76.7% | 69.8% |

#### 8.2.1.2 Concatenated Subjects

**First Experiment** We concatenated the data of all the 20 subjects as single dataset, then we used our classifiers at each of the three regions we mentioned before:

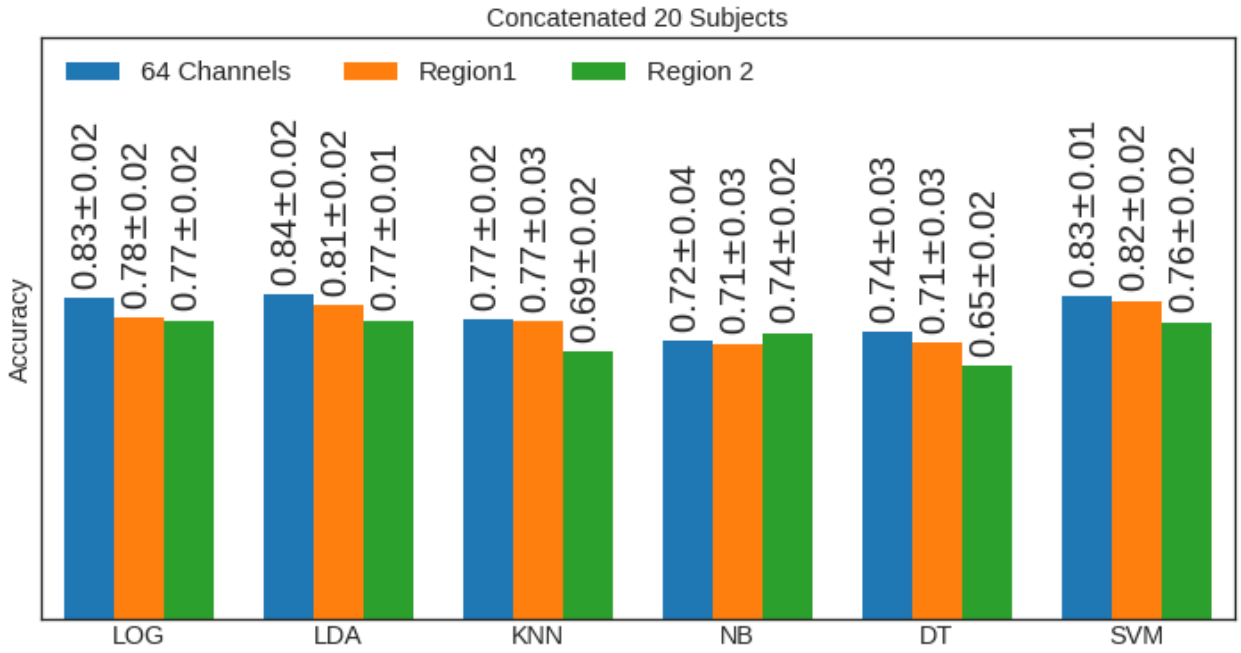



Figure 8.12: This figure shows the mean accuracy and the standard deviation(STD) over 20 subjects concatenated for each classifier.

**Second Experiment** We concatenated the data of the subjects that give high accuracy who are: subject 1, subject 2, subject 3, subject 6, subject 7, subject 9, subject 15, subject 18, subject 20 as single dataset, then we used our classifiers at each of the three regions we mentioned before:

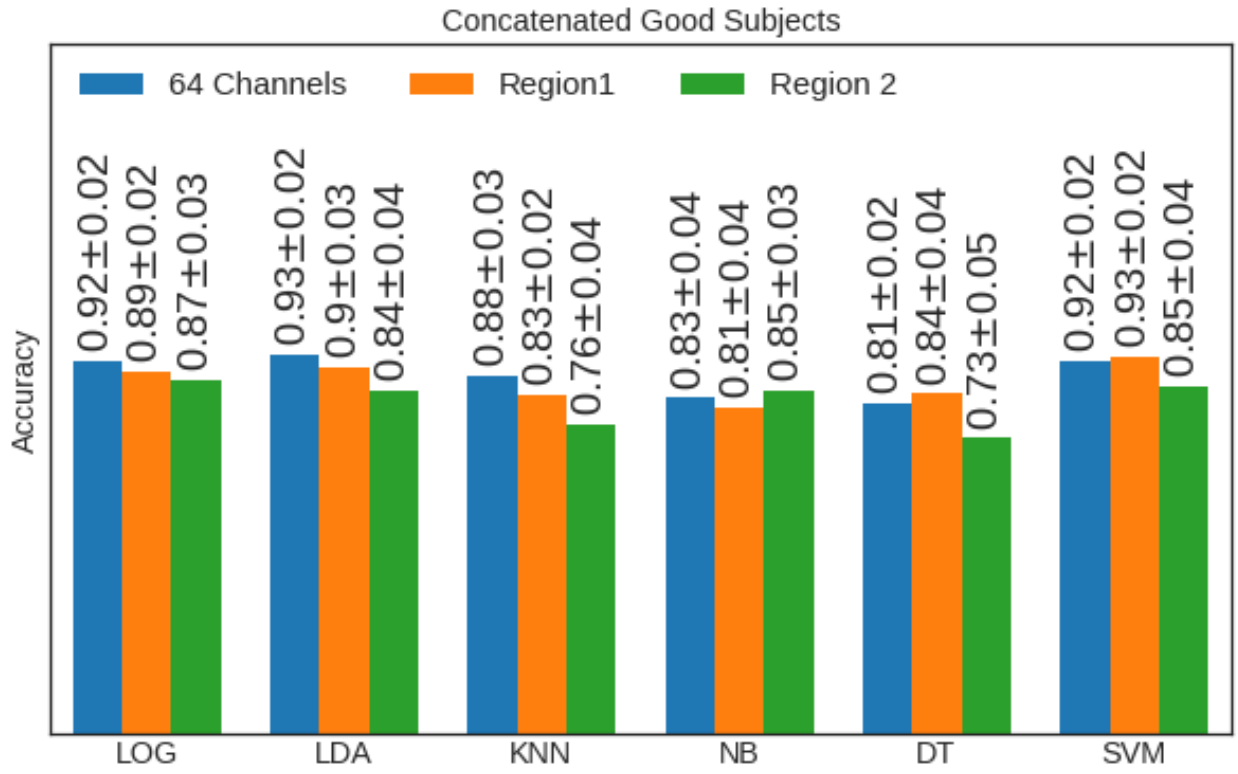



Figure 8.13: This figure shows the mean accuracy and the standard deviation(STD) over selected subjects concatenated for each classifier.

## 8.3 recorded Experiment

In this section we will introduce our online experimentation phase, which contains our recorded data, which was recorded from 16 persons(all males) using the EMOTIV EPOC+ cap we recorded 4 files for each subject contains the recorded signals in EDF file format which we will talk about it later, also we will talk about our scenario of recording that data, then we'll talk about deploying our machine learning algorithms on that data showing the effect of the different algorithms on it and also the different parameters of these algorithms.
Our data is formed of 16 subjects all males between 20-26 years old for each one we recorded four sessions. Each session contains 2 imagination

movements(right hand, left hand) or (tongue, feet) , we record one file for each session and each file contains 20 trials, 10 trials for each imagine of movement.
Our scenario of recording the data, the figure below shows the phases of recording the data:

1. First, baseline time for 10 seconds with closed eyes.
2. Second, beep sound for 1 second to make the subject open his eyes and start the next phase.
3. 10 seconds of baseline with opened eyes.
4. Then, start the imagination phase for 4 seconds.
5. Then, 4 seconds resting.
6. Iterate from step 4 and 5, 20 times shuffled.

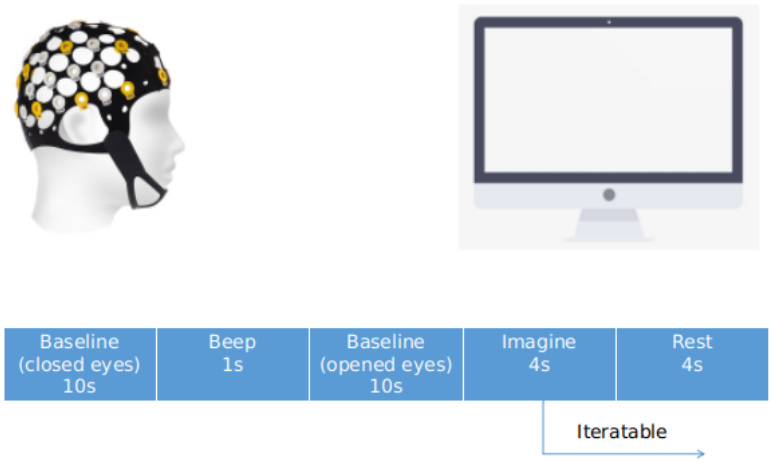



Figure 8.14: This figure shows the describtion of the data.

## 8.3.1 Experiments

### 8.3.1.1 Dependent Subjects

**First Expriment** In this Experiment we used Linear Discriminant Analysis(LDA) with solver least square solution and shrinkage with ledoit-wolf

lemma algorithm. Number of component of CSP in this Experiment was 8 and transform data using fourier transform and take the first 9 frequencies on all channels and also used CSP with 7 components, and 8 components of fourier tranform on the region of channels: ['AF3' , 'F7' , 'F3' , 'FC5' , 'T7', 'P7', 'O1' ,'FC6' , 'F8' ].

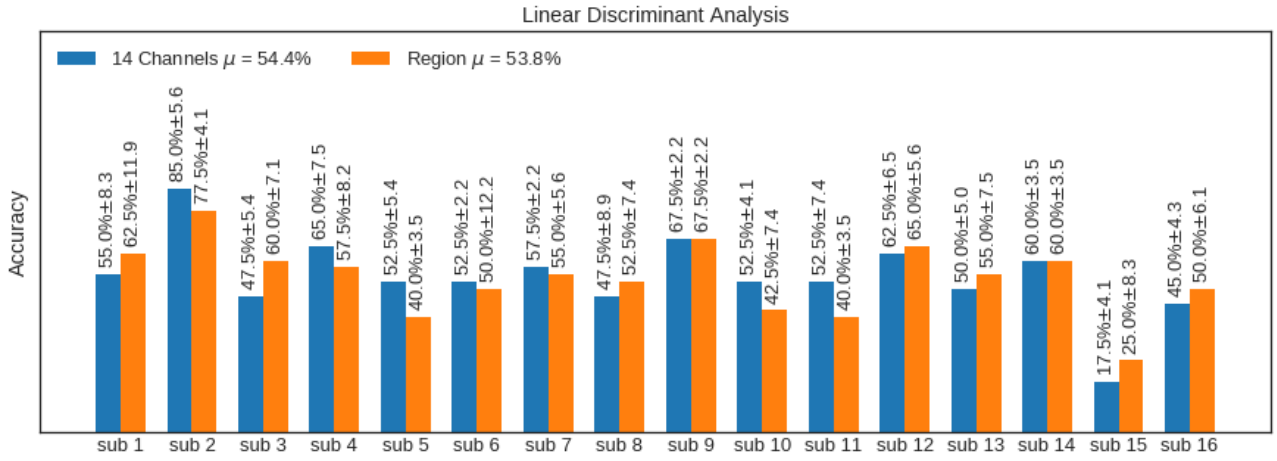



Figure 8.15: This figure shows the performance of LDA algorithm.

**Second Expriment** In this Experiment we used K-Nearest Neighbour(KNN), thenumber of component of CSP in this Experiment was 13 and transform data using fourier transform and take the first 9 frequencies on all channels, and also used ICA with 8 components on the region of channels: ['AF3' , 'F7' , 'F3' , 'FC5' , 'T7', 'P7', 'O1' ,'FC6' , 'F8' ].

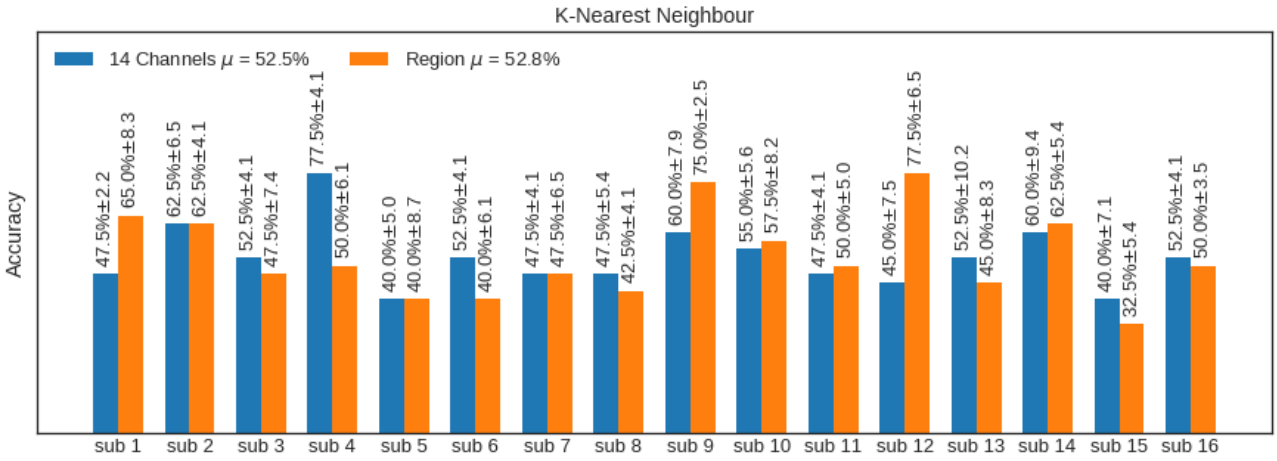



Figure 8.16: This figure shows the performance of KNN algorithm.

**Third Expriment** In this Experiment we used Logistic Regression, the number of componenets of fourier transform was 11 on all channels, and also used CSP with 7 and 8 fourier transform componenets on the region of channels: ['AF3' , 'F7' , 'F3' , 'FC5' , 'T7', 'P7', 'O1' ,'FC6' , 'F8' ].

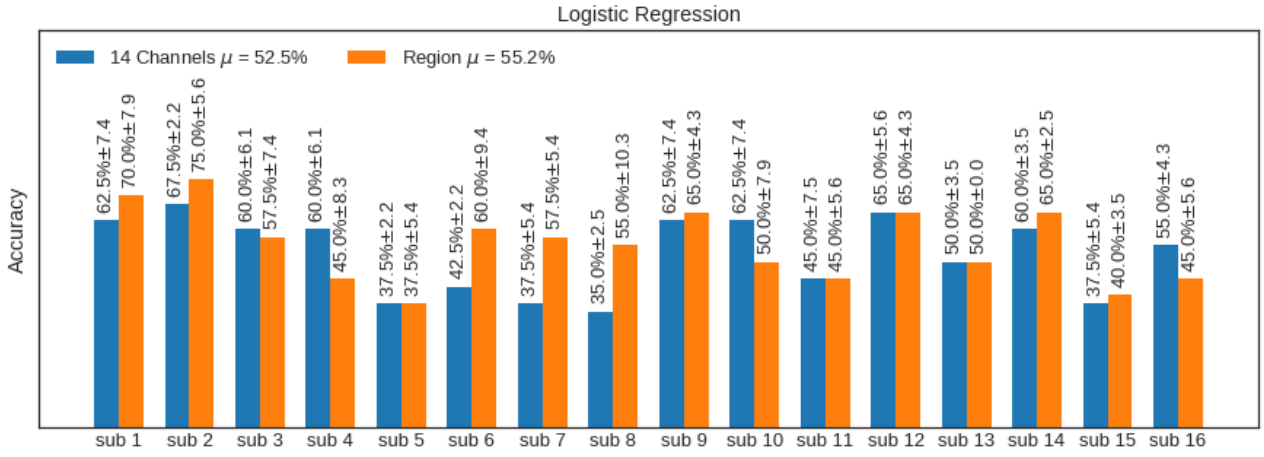



Figure 8.17: This figure shows the performance of LOG algorithm.

**Forth Expriment** In this Experiment we used Naive Bayes(NB), the number of componenets of PCA was 7 on all channels, and also used CSP with

5 componenets on the region of channels: ['AF3' , 'F7' , 'F3' , 'FC5' , 'T7', 'P7', 'O1' ,'FC6' , 'F8' ].

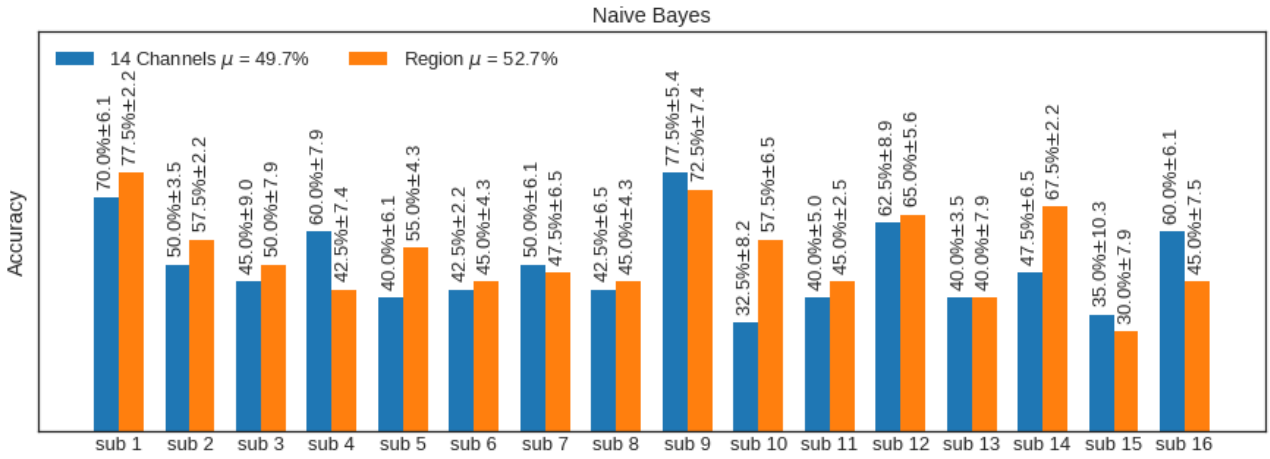



Figure 8.18: This figure shows the performance of Naive Bayes algorithm.

**Forth Expriment** In this Experiment we used Support Vector Machine(SVM), the number of componenets of fourier tranform was 8 on all channels, and also used fourier transfer with 8 componenets on the region of channels: ['AF3' , 'F7' , 'F3' , 'FC5' , 'T7', 'P7', 'O1' ,'FC6' , 'F8' ].

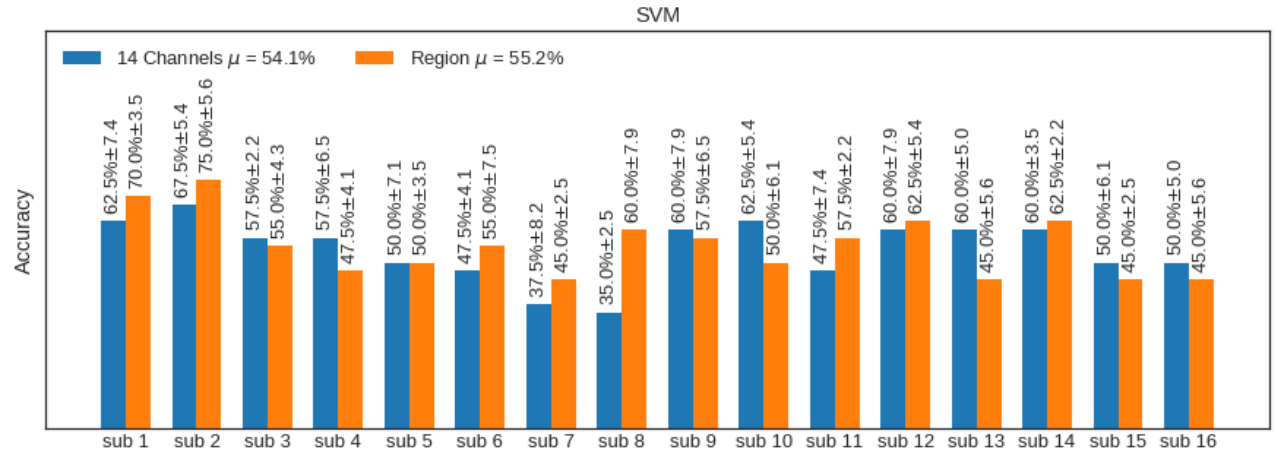



Figure 8.19: This figure shows the performance of SVM algorithm.

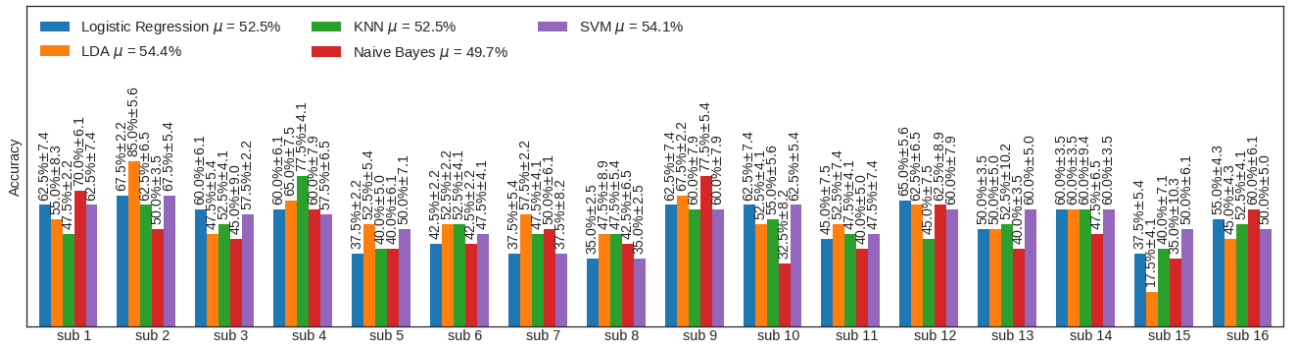



Figure 8.20: This figure shows the performance of all algorithms on all subjects.

| Classification | LOG | LDA | KNN | NB | SVM |
|---|---|---|---|---|---|
| Sub 1 | $62.5\% \pm 7.4$ | $55.0\% \pm 8.3$ | $47.5\% \pm 2.2$ | $70.0\% \pm 6.1$ | $62.5\% \pm 7.4$ |
| Sub 2 | $67.5\% \pm 2.2$ | $85.0\% \pm 5.6$ | $62.5\% \pm 6.5$ | $50.0\% \pm 3.5$ | $67.5\% \pm 5.4$ |
| Sub 3 | $60.0\% \pm 6.1$ | $47.5\% \pm 5.4$ | $52.5\% \pm 4.1$ | $45.0\% \pm 9.0$ | $57.5\% \pm 2.2$ |
| Sub 4 | $60.0\% \pm 6.1$ | $65.0\% \pm 7.5$ | $77.5\% \pm 4.1$ | $60.0\% \pm 7.9$ | $57.5\% \pm 6.5$ |
| Sub 5 | $37.5\% \pm 2.2$ | $52.5\% \pm 5.4$ | $40.0\% \pm 5.0$ | $40.0\% \pm 6.1$ | $50.0\% \pm 7.1$ |
| Sub 6 | $42.5\% \pm 2.2$ | $52.5\% \pm 2.2$ | $52.5\% \pm 4.1$ | $42.5\% \pm 2.2$ | $47.5\% \pm 4.1$ |
| Sub 7 | $37.5\% \pm 5.4$ | $57.5\% \pm 2.2$ | $47.5\% \pm 4.1$ | $50.0\% \pm 6.1$ | $37.5\% \pm 8.2$ |
| Sub 8 | $35.0\% \pm 2.5$ | $47.5\% \pm 8.9$ | $47.5\% \pm 5.4$ | $42.5\% \pm 6.5$ | $35.0\% \pm 2.5$ |
| Sub 9 | $62.5\% \pm 7.4$ | $67.5\% \pm 2.2$ | $60.0\% \pm 7.9$ | $77.5\% \pm 5.4$ | $60.0\% \pm 7.9$ |
| Sub 10 | $62.5\% \pm 7.4$ | $52.5\% \pm 4.1$ | $55.0\% \pm 5.6$ | $32.5\% \pm 8.2$ | $62.5\% \pm 5.4$ |
| Sub 11 | $45.0\% \pm 7.5$ | $52.5\% \pm 7.4$ | $47.5\% \pm 4.1$ | $40.0\% \pm 5.0$ | $47.5\% \pm 7.4$ |
| Sub 12 | $65.0\% \pm 5.6$ | $62.5\% \pm 6.5$ | $62.5\% \pm 7.5$ | $45.0\% \pm 7.5$ | $62.5\% \pm 8.9$ |
| Sub 13 | $50.0\% \pm 3.5$ | $50.0\% \pm 5.0$ | $52.5\% \pm 10.2$ | $40.0\% \pm 3.5$ | $60.0\% \pm 5.0$ |
| Sub 14 | $60.0\% \pm 3.5$ | $60.0\% \pm 3.5$ | $60.0\% \pm 9.4$ | $47.5\% \pm 6.5$ | $60.0\% \pm 3.5$ |
| Sub 15 | $37.5\% \pm 5.4$ | $17.5\% \pm 4.1$ | $40.0\% \pm 7.1$ | $35.0\% \pm 10.3$ | $50.0\% \pm 6.1$ |
| Sub 16 | $55.0\% \pm 4.3$ | $45.0\% \pm 4.3$ | $52.5\% \pm 4.1$ | $60.0\% \pm 6.1$ | $50.0\% \pm 5.0$ |
| Mean | 52.5% | 54.4% | 52.5% | 49.7% | 54.1% |

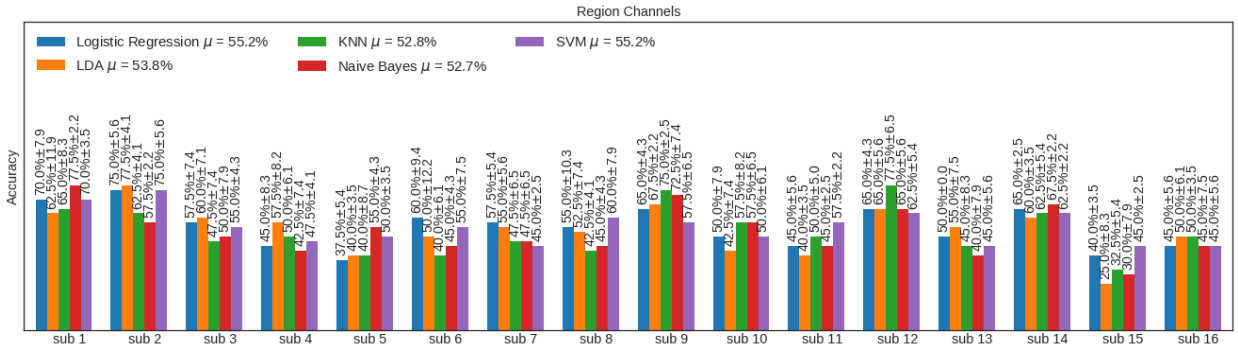



Figure 8.21: This figure shows the performance of all algorithms on all subjects.

| Classification | LOG | LDA | KNN | NB | SVM |
|---|---|---|---|---|---|
| Sub 1 | 70.0% ± 7.9 | 62.5% ± 11.9 | 65.0% ± 8.3 | 77.5% ± 2.2 | 70.0% ± 3.5 |
| Sub 2 | 75.0% ± 5.6 | 77.5% ± 4.1 | 62.5% ± 4.1 | 57.5% ± 2.2 | 75.0% ± 5.6 |
| Sub 3 | 57.5% ± 7.4 | 60.0% ± 7.1 | 47.5% ± 7.4 | 50.0% ± 7.9 | 55.0% ± 4.3 |
| Sub 4 | 45.0% ± 8.3 | 57.5% ± 8.2 | 50.0% ± 6.1 | 42.5% ± 7.4 | 47.5% ± 4.1 |
| Sub 5 | 37.5% ± 5.4 | 40.0% ± 3.5 | 40.0% ± 8.7 | 55.0% ± 4.3 | 50.0% ± 3.5 |
| Sub 6 | 60.0% ± 9.4 | 50.0% ± 12.2 | 40.0% ± 6.1 | 45.0% ± 4.3 | 55.0% ± 7.5 |
| Sub 7 | 57.5% ± 5.4 | 55.0% ± 5.6 | 47.5% ± 6.5 | 47.5% ± 6.5 | 45.0% ± 2.5 |
| Sub 8 | 55.0% ± 10.3 | 52.5% ± 7.4 | 42.5% ± 4.1 | 45.5% ± 4.3 | 60.0% ± 7.9 |
| Sub 9 | 65.0% ± 4.3 | 67.5% ± 2.2 | 75.0% ± 2.5 | 72.5% ± 7.4 | 57.5% ± 6.5 |
| Sub 10 | 50.0% ± 7.9 | 42.5% ± 7.4 | 57.5% ± 8.2 | 57.5% ± 6.5 | 50.0% ± 6.1 |
| Sub 11 | 45.0% ± 5.6 | 40.0% ± 3.5 | 50.0% ± 5.0 | 45.0% ± 2.5 | 57.5% ± 2.2 |
| Sub 12 | 65.0% ± 4.3 | 65.0% ± 5.6 | 77.5% ± 6.5 | 65.0% ± 5.6 | 62.5% ± 5.4 |
| Sub 13 | 50.0% ± 0.0 | 55.0% ± 7.5 | 45.0% ± 8.3 | 40.0% ± 7.9 | 45.0% ± 5.6 |
| Sub 14 | 65.0% ± 2.5 | 60.0% ± 3.5 | 62.5% ± 5.4 | 67.5% ± 2.2 | 62.5% ± 2.2 |
| Sub 15 | 40.0% ± 3.5 | 25.0% ± 8.3 | 32.5% ± 5.4 | 30.0% ± 7.9 | 45.0% ± 2.5 |
| Sub 16 | 45.0% ± 5.6 | 50.0% ± 6.1 | 50.0% ± 3.5 | 45.0% ± 7.5 | 45.0% ± 5.6 |
| Mean | 55.2% | 53.8% | 52.8% | 52.7% | 55.2% |

### 8.3.1.2 Concatenated Subjects

# Chapter 9

# Conclusion

After we get our data we worked on its signals as a Time Domain then we found that if we worked on it as Frequency Domain that's make deferent in the accuracy.

We concluded that when we used PCA with FFT in the preprocessing phase with LDA classifier, that the best algorithm through our multiple algorithms that we used, and after a lot of experiments on LDA with a lot of tuning of his parameters, we found that it is the best algorithm with this parameters:

- with a solver least square solution and shrinkage with ledoit-wolf lemma algorithm.
- Number of components of PCA in this experiment is 13.
- transform data using Fourier Transform and take the first 10 frequencies (Region 1 and 64 channels).
- We got mean accuracy 86.4% on 10 subjects for 64 channels.
- We got mean accuracy 83.5% on 10 subjects for 15 channels.

Then we when used Gaussian Naïve Bayes with Fourier transform and take the first 10 frequencies and the result was mean accuracy 83.5% on 10 subjects 78.2% for 6 channels.

In another structure of system, we concatenated data of 20 subjects to work with it as one subject, the best result on 64 channels come from LDA with the same parameters and preprocessing in the previous experiment and got mean accuracy 84%, and got best accuracy on 6 channels 77%.

In 15 channels, we got high accuracy from SVM using ICA with number of components 6, and FFT 10 frequencies 82%.

Then we used another structure we concatenated the data of the subjects that give high accuracy who are: Subject1,2,3,6,7,9,15,18,20 as single database, and we got high accuracy on SVM on 15 channels with ICA with number of components 6, and FFT 10 frequencies 93%. And in 6 channels we got high accuracy on Logistic regression with ICA with the number of components 9 and FFT 10 frequencies 84%.

Finally, we recorded data which was recorded from 16-person (all males) we used LDA classifier with solver least square solution and shrinkage with legoit-wolf lemma algorithm and CSP with number of components 8 and FFT 9 frequencies for all channels (14 channels) and got mean (for 16 subject) accuracy 54.4%. And we used Logistic regression with CSP with 7 components and FFT with 8 frequencies on 9 channels we got 55.2% accuracy.

# Chapter 10

# Future Work

Future Work:

- improve accuracy.
  We will try to make our accuracy higher than we got
- Hardware component.
  we want to make our project more helpful so we will turn it to product can help people so we will add hardware like a hand to help the paralyzed people to contact with their world.
- live streaming.
  it will be done by making our project work in the real time.
- More Movement.
  We will make our project more usable so we will try to make more movement like high, down, right leg and left leg.
- Try more models on independent architecture.
  we will work more on the independent architecture on our data

# Bibliography


[1] J. B. Park and J. D. Bronzino, "The biomedical engineering handbook," *Boca Raton, FL: CRC Press*, vol. 4, pp. 1–8, 2000.

[2] J. Carey, "Brain facts: A primer on the brain and nervous system.," 1990.

[3] E. R. Kandel, J. H. Schwartz, T. M. Jessell, D. of Biochemistry, M. B. T. Jessell, S. Siegelbaum, and A. Hudspeth, *Principles of neural science*, vol. 4. McGraw-hill New York, 2000.

[4] R. Rohkamm and S. O. Wandrey, *Color atlas of neurology*. Thieme, 2004.

[5] N. S. Malan and S. Sharma, "Feature selection using regularized neighbourhood component analysis to enhance the classification performance of motor imagery signals," *Computers in Biology and Medicine*, vol. 107, pp. 118 – 126, 2019.

[6] A. Singh, S. Lal, and H. W. Guesgen, "Reduce calibration time in motor imagery using spatially regularized symmetric positives-definite matrices based classification," *Sensors (Basel, Switzerland)*, vol. 19, January 2019.

[7] H. Fauzi, M. I. Shapiai, and U. Khairuddin, "Transfer learning of bci using cur algorithm," *Journal of Signal Processing Systems*, Feb 2019.

[8] X. Zhu, P. Li, C. Li, D. Yao, R. Zhang, and P. Xu, "Separated channel convolutional neural network to realize the training free motor imagery

bci systems," *Biomedical Signal Processing and Control*, vol. 49, pp. 396 – 403, 2019.

[9] D. C. Irimia, R. Ortner, M. S. Poboroniuc, B. E. Ignat, and C. Guger, "High classification accuracy of a motor imagery based brain-computer interface for stroke rehabilitation training," *Frontiers in Robotics and AI*, vol. 5, p. 130, 2018.

[10] S. Sanei and J. Chambers, *EEG SIGNAL PROCESSING*. 2007.

[11] G. Gobbi, F. Bouquet, L. Greco, A. Lambertini, C. Tassinari, A. Ventura, and M. Zaniboni, "Coeliac disease, epilepsy, and cerebral calcifications," 1992.

[12] R. Bickford, *Electroencephalography: encyclopedia of neuroscience*. 1987.

[13] Z. Koles, "The quantitative extraction and topographic mapping of the abnormal components in the clinical eeg," 1991.

[14] E. C. Cherry, "Some experiments in the recognition of speech, with one and two ears," 1953.

[15] C. M. B. Michael E. Tipping, "Mixtures of probabilistic principal component analysers," 1999.