University of Denver
Digital Commons @ DU

Electronic Theses and Dissertations | Graduate Studies

1-1-2019

Applied Machine Learning for Classification of Musculoskeletal Inference using Neural Networks and Component Analysis

Shaswat Sharma
University of Denver

Follow this and additional works at: https://digitalcommons.du.edu/etd

 Part of the Artificial Intelligence and Robotics Commons

Recommended Citation

Sharma, Shaswat, "Applied Machine Learning for Classification of Musculoskeletal Inference using Neural Networks and Component Analysis" (2019). *Electronic Theses and Dissertations*. 1619.
https://digitalcommons.du.edu/etd/1619

This Thesis is brought to you for free and open access by the Graduate Studies at Digital Commons @ DU. It has been accepted for inclusion in Electronic Theses and Dissertations by an authorized administrator of Digital Commons @ DU. For more information, please contact jennifer.cox@du.edu,dig-commons@du.edu.

Applied Machine Learning for Classification of Musculoskeletal Inference using Neural Networks and Component Analysis

Abstract

Artificial Intelligence (AI) is acquiring more recognition than ever by researchers and machine learning practitioners. AI has found significance in many applications like biomedical research for cancer diagnosis using image analysis, pharmaceutical research, and, diagnosis and prognosis of diseases based on knowledge about patients' previous conditions. Due to the increased computational power of modern computers implementing AI, there has been an increase in the feasibility of performing more complex research.

Within the field of orthopedic biomechanics, this research considers complex time-series dataset of the "sit-to-stand" motion of 48 Total Hip Arthroplasty (THA) patients that was collected by the Human Dynamics Laboratory at the University of Denver. The research focuses on predicting the motion quality of the THA patients by analyzing the loads acting on muscles and joints during one motion cycle. We have classified the motion quality into two classes: "Fair" and "Poor", based on muscle forces, and have predicted the motion quality using joint angles.

We address different types of Machine Learning techniques: Artificial Neural Networks (LSTM - long short-term memory, CNN - convolutional neural network, and merged CNN-LSTM) and data science approach (principal component analysis and parallel factor analysis), that utilize remodeled datasets: heatmaps and 3-dimensional vectors. These techniques have been demonstrated efficient for the classification and prediction of the motion quality.

The research proposes time-based optimization by predicting the motion quality at an initial stage of musculoskeletal model simulation, thereby, saving time and efforts required to perform multiple model simulations to generate a complete musculoskeletal modeling dataset. The research has provided efficient techniques for modeling neural networks and predicting post-operative musculoskeletal inference. We observed the accuracy of 83.33% for the prediction of the motion quality under the merged LSTM and CNN network, and autoencoder followed by feedforward neural network. The research work not only helps in realizing AI as an important tool for biomedical research but also introduces various techniques that can be utilized and incorporated by engineers and AI practitioners while working on a multi-variate time-series wide shaped data set with high variance.

Document Type

Thesis

Degree Name

M.S.

Department

Computer Science

First Advisor

Matthew J. Rutherford, Ph.D.

Second Advisor

Laleh Mehran, M.F.A.

Third Advisor
Rinku Dewri

Keywords
Cross validation, Deep learning, Motion quality, Neural network, Quality of motion, Total hip arthroplasty

Subject Categories
Artificial Intelligence and Robotics | Computer Sciences

Publication Statement
Copyright is held by the author. User is responsible for all copyright compliance.

This thesis is available at Digital Commons @ DU: https://digitalcommons.du.edu/etd/1619

Applied Machine Learning for Classification of Musculoskeletal Inference using Neural Networks and Component Analysis

---

A Thesis
Presented to
the Faculty of the
Daniel Felix Ritchie School of
Engineering and Computer Science
University of Denver

In Partial Fulfillment
of the Requirements for the Degree
Master of Science

---

by
Shaswat Sharma
June 2019
Advisor: Matthew Rutherford

©Copyright by Shaswat Sharma 2019
All Rights Reserved

Author: Shaswat Sharma
Title: Applied Machine Learning for Classification of Musculoskeletal Inference using Neural Networks and Component Analysis
Advisor: Matthew Rutherford
Degree Date: June 2019

Abstract

Artificial Intelligence (AI) is acquiring more recognition than ever by researchers and machine learning practitioners. AI has found significance in many applications like biomedical research for cancer diagnosis using image analysis, pharmaceutical research, and, diagnosis and prognosis of diseases based on knowledge about patients' previous conditions. Due to the increased computational power of modern computers implementing AI, there has been an increase in the feasibility of performing more complex research.

Within the field of orthopedic biomechanics, this research considers complex time-series dataset of the "sit-to-stand" motion of 48 Total Hip Arthroplasty (THA) patients that was collected by the Human Dynamics Laboratory at the University of Denver. The research focuses on predicting the motion quality of the THA patients by analyzing the loads acting on muscles and joints during one motion cycle. We have classified the motion quality into two classes: "Fair" and "Poor", based on muscle forces, and have predicted the motion quality using joint angles.

We address different types of Machine Learning techniques: Artificial Neural Networks (LSTM - long short-term memory, CNN - convolutional neural network, and merged CNN-LSTM) and data science approach (principal component analysis and parallel factor analysis), that utilize remodeled datasets: heatmaps and 3-dimensional vectors. These techniques have been demonstrated efficient for the classification and prediction of the motion quality.

The research proposes time-based optimization by predicting the motion quality at an initial stage of musculoskeletal model simulation, thereby, saving time and efforts required to perform multiple model simulations to generate a complete musculoskeletal modeling dataset. The research has provided efficient techniques for modeling neural networks and predicting post-operative musculoskeletal inference. We observed

the accuracy of 83.33% for the prediction of the motion quality under the merged LSTM and CNN network, and autoencoder followed by feedforward neural network. The research work not only helps in realizing AI as an important tool for biomedical research but also introduces various techniques that can be utilized and incorporated by engineers and AI practitioners while working on a multi-variate time-series wide shaped data set with high variance.

Acknowledgements

It gives me great pleasure to thank all the important people who have helped me during the thesis, without whom, thesis completion would have been next to impossible.

My heartiest thanks to my parents, Jagdish and Jyotsna, and my brother, Ateet, for their endless love and faith in me that always inspired me to work harder.

I want to express my sincere gratitude towards my research advisor, Dr. Matthew Rutherford, who has been very generous in providing me the knowledge and his time for troubleshooting and brainstorming. While working on the research, he scheduled numerous meetings with me and always guided me in the right direction; applying that in the research work produced tremendously improved and accurate results. I could not think of any better mentor than Dr. Matt.

I would also like to thank my defense committee members, Rinku Dewri and Laleh Mehran, for reviewing my thesis and for being part of my oral defense committee. Their valuable insights have helped me incredibly in delivering the work.

The University of Denver is a wonderful school that has always maintained an accomplishing sentiment in my heart. I feel proud to be a graduate student at the school where I have learned a great deal of science.

Last but not the least, I would like to thank my friends and colleagues for their continuous encouragement.

Table of Contents

List of Figures

Chapter 1

Introduction

Machine Learning (ML), a subfield of Artificial Intelligence (AI), is the field of computer science that attempts to mimic the cognitive functions like learning and problem solving that a human brain does. With the help of AI techniques, machines have been successful in demonstrating intelligent behavior imitating human intelligence. Based on past experience, the machines can make decisions on a new set of information that represents the analytical ability to respond to a new environment. Some of the practical applications of AI are natural language processing like Siri, Alexa and Cortana, image analysis for object recognition, medical diagnosis, time-series predictions for stock markets, and so on. Recently, AI is powered by Artificial Neural Networks (ANN) to analyze big data and enable such applications.

ANN is a self-learning programming framework loosely inspired by the biological organization of animal brains. The design of ANN is such that they are meant to replicate the way humans learn. ANN aid the computing systems to assimilate and learn using observational data and ML algorithms. The ANN model performs tasks by "learning" from a set of input-output examples, rather than being programmed with a set of rules. The accuracy of a neural network depends on multiple factors, one of which is the size of the dataset (number of input-output examples) available as input to the network. A larger dataset gives more information on the input test cases, and hence, it enables robust learning.

Due to the increasing availability of data, AI has also flourished in the field of medical science by helping doctors and clinicians improve different aspects of health care like prevention of diseases, providing effective treatments to patients, disease prognosis, pre-operative and post-operative care of surgical procedures and pharmaceutical drugs, among others. The availability of a patient's medical history and previous medical cases allows systems to extract knowledge and analyze future instances. In the biomedical orthopedic industry, scientists are using the latest technologies extensively to improve orthopedics with robot-assisted surgery, improved designs of prosthetic implants, shoe designing for different purposes for the best comfort of users, and many more.

Human dynamics, a branch of orthopedic biomechanics, focuses on studying the human musculoskeletal system to improve clinical diagnosis and treatment through biomechanical measurement and analysis. A human musculoskeletal system encompasses the organ system that gives humans the ability to perform movements using the muscular and skeletal systems. It consists of joints, bones, muscles, ligaments, cartilage, and various other connective tissues. The availability of software for biomechanical modeling, simulation, and analysis has helped in improving the study of human dynamics with efficient data collection of the human body movements [3].

Human motion analysis of a subject's gait, sit-to-stand, and step-down movements has been useful in the field of sports, kinesiology, physical therapy, and surgical analysis. Surgeries like knee arthroplasty, hip joint replacement, laminectomy, and spinal fusion are the consequences of reduced mobility of human body movements and are common in United States [1]. Motion analysis has enabled clinicians to assess and treat patients with deficiencies in body movement. Motion discrepancy in the musculoskeletal system can be observed at a fine level of detail with the help of advanced machinery that captures the motion of patients, which is further analyzed by simulating a virtual musculoskeletal model. Quantifying the human motion into joint and muscle forces helps in assessing post-operative impacts on the patients.

Total Hip Arthroplasty (THA) or hip replacement surgery is a common surgical procedure used for the treatment of joint failure caused by osteoarthritis. The hip joint is the amalgamation of the femoral head (the highest part of the thigh bone) and acetabulum (cavity where femoral head and pelvis meet). THA involves the substitu-

tion of both femoral head and acetabulum with prosthetic implants. The alignment of the three synthetic components significantly influences post-surgical motion quality [4]. To study the component alignment for the hip replacement surgery, scientists and researchers in the Human Dynamics Laboratory (HDL) at the University of Denver have collected the musculoskeletal data of THA patients using equipment like force platforms, electronic markers, motion capture systems, and OpenSim software. We use the same musculoskeletal dataset to assess the motion quality of THA patients with ML.

1.1 Problem Statement

This research addresses the application of machine learning and data science in the field of orthopedic biomechanics by utilizing various ANN architectures and applying data science approaches. The work comprises of the quantitative analysis of the THA patients' body movements and assessment of their motion quality, based on data science and ML models. The resulting motion quality metric can help modify the component alignment to improve the results of a surgical procedure and enable improved prosthetic implantation.

We have used the musculoskeletal data quantified by the HDL for the post-surgical "sit-to-stand" motion of 48 THA patients. The dataset contains variables corresponding to skeletal joints (pelvis, hip, knee, ankle and lumbar) and muscles (biceps femoris, semimembranosus, semitendinosus, gluteus, rectus, vastus, gastrocnemius, and soleus) of the lower extremity. Thus, the attributes of the patient dataset entirely correspond to the bio-medical field and are more understandable to the doctors and the clinicians. We, being the researchers in the field of Computer Science, attempt to comprehend the data and make inferences using ML techniques. The dataset has 48 records each with 7502 columns, with 75 time-series features, each having 100 time units. However, standard ML algorithms prefer many thousands of records, and to account for that, we have analyzed the data efficiently by applying various ANN models.

The motion quality of THA patients is quantified using data science approaches like principal component analysis and parallel factor analysis, and it is classified into two classes: "Fair" and "Poor". The research proposes to predict the quantified motion

quality from the first stage of data simulation, inverse kinematics, instead of simulating the variables of the other two consecutive stages - inverse dynamics and muscle force prediction. The motion quality is thus, predicted from time-series variables of inverse kinematics using ANN models like feedforward, CNN (Convolutional neural network), LSTM (Long short-term memory), and auto-encoders.

1.2 Summary of Contributions

The research work has addressed various methods, techniques, and approaches, for engineers and researchers engaged in the field of machine learning. The contributions can be summarized as follows:

- **Motion quality quantification:** This work discusses how we use the THA patients' dataset and remodel it by applying various techniques to quantify the motion quality. As there is no standard way to assess the motion quality from the available muscle force attributes of the dataset, we have devised a method for relative analysis between the patients that can help us quantify the motion quality.
- **Motion quality discovery:** Using component analysis techniques, we discovered the existence of quantified motion quality in the joint angle dataset of the lower extremity. Further, we used the results of the analysis for inverse kinematics feature selection and to improve the motion quality quantification.
- **Shortened process to predict motion quality:** This research describes the application of various ANN models to predict the quantified motion quality from the time-series variables of inverse kinematics (the first stage of musculoskeletal data simulation). It eliminates the need to calculate the variables of inverse dynamics and muscle force prediction.
- **Wide-shaped and high variance data utilization:** The dataset used is wide-shaped and has high variance; such a dataset is considered unreliable in machine learning. This work demonstrates efficient data science techniques and neural network models for such an input dataset.
- **Remodeled time-series data analysis:** We remodeled the time-series dataset into 2-dimensional and 3-dimensional vectors and used them in two different component analysis techniques for the discovery of motion quality in the joint

angle dataset. Also, we used the remodeled datasets along with heatmaps to train the neural network models to learn and classify the motion quality of each patient efficiently.

- **Motion quality prediction:** We have predicted the classified motion quality via muscle forces, using the joint angles by ANN, with an accuracy of 83.33%.

1.3 Organization of Thesis

The thesis work is structured around the following chapters:

- **Related Work and Background:** This chapter explores the previous research work and relevant contributions made in the field of ANN and musculoskeletal modeling. It also describes various ANN models and data science approaches, and concepts related to them, that are used in this research.
- **System Description:** This chapter encompasses the data collection process and data normalization techniques. It has a description of the data simulation stages for musculoskeletal modeling viz inverse kinematics, inverse dynamics, and muscle force predictions. It also focuses on narrowing the body features that are most relevant to the motion quality analysis. The chapter also presents the relative motion analysis of one patient against other patients.
- **Research and Analysis:** This chapter constitutes the primary research work and analysis that builds this thesis. It consists of the following:
  - Motion quality analysis, quantification, classification, and verification.
  - Application of the component analysis to get the most concentrated and relevant information from the wide-shaped dataset, achieved by data remodeling.
  - Motion quality evolution by selection of relevant features for prediction and classification.
  - Assessment of conventional regression methods over the dataset in application.
  - Exploration and application of ANN models to perform data compression and prediction of motion quality, using different remodeled datasets like heatmaps, time-series, 2-dimensional and 3-dimensional vectors.

  - Development of good performance networks to achieve high accuracy for motion quality prediction.

- **Conclusions and Future Work:** This chapter summarizes the contributions of the research work and talks about the ways to explore this work further.

Chapter 2

Related Work and Background

2.1 Total Hip Arthroplasty

The hip joint is analogous to a ball-socket joint where the spherical ball lying in the socket enables the rotation and multi-directional movement of the coupling. In a hip joint, the femoral head (top of the thigh bone) acts as the ball, and the acetabulum (cavity where femoral head and pelvis meets) serves as the socket, allowing the smooth movement of the hip joint [20]. Figure 2.1 depicts the hip joint and its components.

Hip joint deterioration can be caused by osteoarthritis A.2.1, inflammatory arthritis A.2.2, infancy hip disorders, trauma and osteonecrosis A.2.3. Non-surgical treatments like physical therapy, weight loss, using a walker, medications or steroid injections are recommended to treat a bad hip before resorting to the surgical procedure of Total Hip Arthroplasty (THA).

In the procedure of THA, prosthetic implants made from metal or plastic replace the deteriorated hip joint. The femoral head and the acetabulum are replaced with a prosthetic femoral component and an acetabular cup respectively. The femoral component is made of the femoral head and femoral stem. In the surgical procedure, the acetabular cup is fitted with the femoral component with a plastic liner to enable a smooth, realistic movement of the prosthetic hip joint. Figure 2.2 depicts the post-surgical view of the hip joint along with the prosthetic components.

Hip replacement surgeries have been one of the most common orthopedic surgical operations in the United States with a reliable success rate [16]. As THA involves

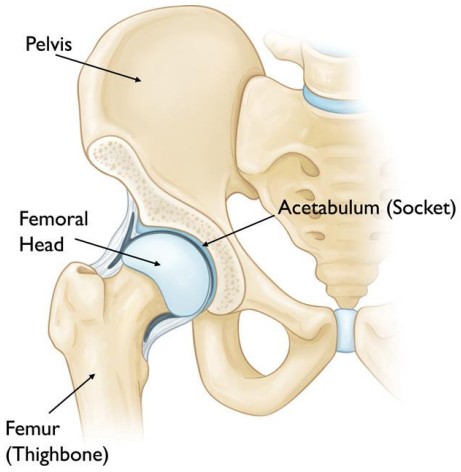


Figure 2.1: Total Hip Arthroplasty : Before surgery [1]

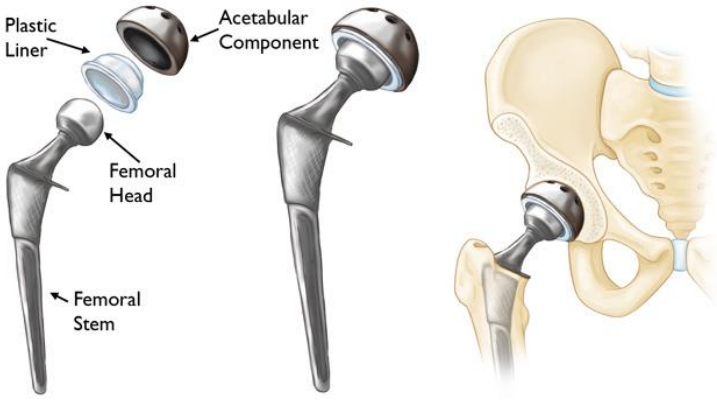


Figure 2.2: Total Hip Arthroplasty : After surgery [1]

the replacement of the hip with prosthetic implantation, the component alignment significantly impacts the post-surgical quality of motion. Component mishaps like impingement, dislocation, component degradation, and edge loading, affect the component alignment which consecutively incurs changes in femoral anteversion and increases hip joint contact forces [17]. Such issues lead to a revision THA which is undesirable. Inferring knowledge about the post-surgical quality of motion can be beneficial to reduce the number of revision THA, and also perform the first THA in other patients with more precision.

[1] https://orthoinfo.aaos.org/

Machine learning techniques have been used for the optimization of joint replacements. Cilla et al. [41] used the following two techniques to solve the prosthetic implant optimization problem:

- Artificial Neural Networks (ANN)
- Support Vector Machines (SVM)

They investigated the geometry of a commercial short stem hip prosthesis using the following implant geometrical parameters:

- Total stem length
- Thickness in the lateral and medial stem sides
- Distance between the central stem surface and the implant neck

The machine learning techniques they used on these parameters showed that optimization in the prosthetic implant is significantly dependent on the decrease in stem length and reduction in the length of the surface in contact with the bone.

2.2 Motion Capture

The process of capturing the human body movements into a digital environment, using motion capture equipment like high-resolution cameras and electronic markers (sensors) is known as motion capture. The process of motion capture involves recording the motion using marker placements, transforming the data into digital form and integrating the data using analytical software.

There are three types of motion capture systems:

1. Magnetic: Sensor-based system where sensors are placed on different body parts. The sensors are wired to a computer that fetch the data corresponding to marker coordinates.
2. Mechanical: Motion is captured using a skeleton-shaped body suit. Each joint is attached with a sensor that gives the relative coordinates of the sensors.
3. Optical: Motion is captured digitally using reflective and pulse LEDs. Multiple cameras are placed at different angles in the environment to create a sense of three dimensions. The sensors are placed on the body, and their positions are mapped into a 3D environment using a computer software.

Researchers working with the Human Dynamics Lab at the University of Denver have used an optoelectronic sensor-based motion capture system to record the "sit-to-stand" motion of THA patients. The sensors attached to the body are called markers. Figure 2.4 gives an overview of the working of the marker-based motion capture system.

Vicon motion capture [13] is one of the most precise marker based optoelectronic motion capture system with a system error of less than 2mm (between actual marker position and software generated marker coordinates) [19]. During the process of motion capture, each body movement is captured across multiple time frames (many times per second) by the electronic markers placed at different places on the body. The markers, acting as sensors, broadcast the captured data to the motion capture software which then generates a 3-dimensional simulated skeleton model corresponding to the body motion.

2.3 Musculoskeletal Modeling

The musculoskeletal models are computational models simulating the human skeletal system that are composed of rigid bodies replicating the bones bridged by mechanical joints. Actuators represent the muscles to provide joint torques mirroring the body movement. Figure 2.3 depicts a simulated musculoskeletal model for three types of motion: gait, step down and sit-to-stand.

External forces on the human body have an impact on the internal muscle loads. To observe the relation between the human body movements and loads on the muscles during the movements, motion analysis is performed using musculoskeletal models. With the advancement in technology, the external forces are quantifiable with the force platforms (instruments to measure the ground reaction forces generated by the motion of the body) but quantifying internal muscle loads, and joint reaction forces are not feasible due to its invasiveness. Musculoskeletal models have been introduced to reach an almost accurate estimation of the parameters.

Musculoskeletal Simulation

The marker-based motion capture system records the human motion and transforms it into a musculoskeletal model with the motion capture software. The ex-

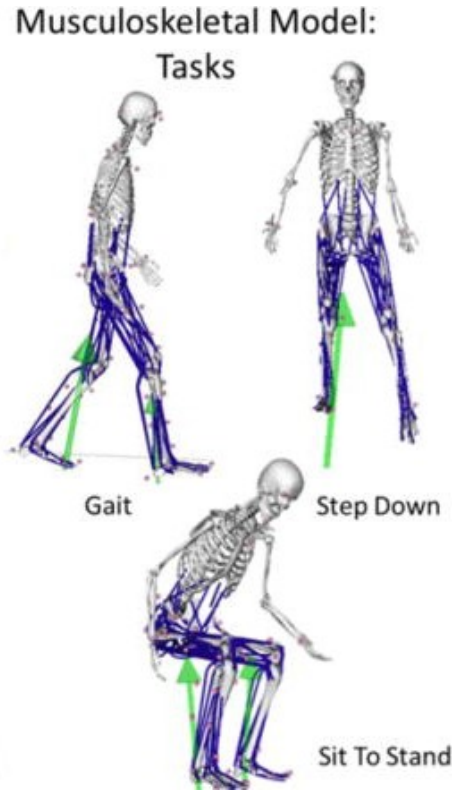


Figure 2.3: Types of motion [23]

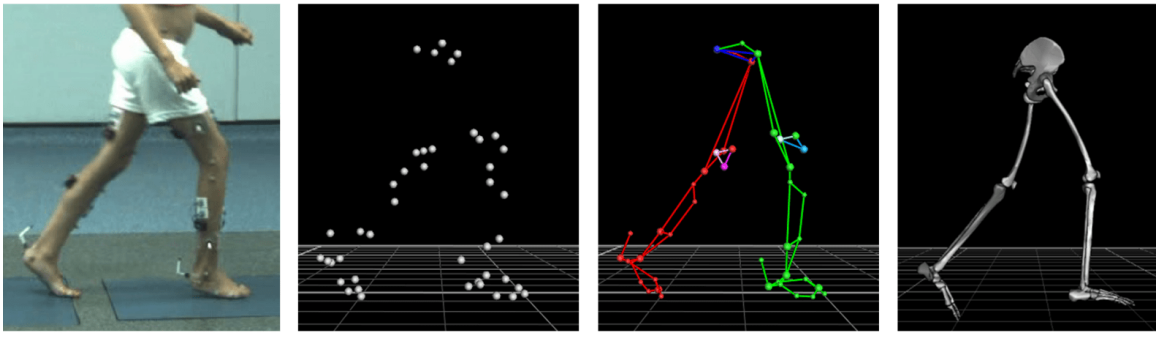


Figure 2.4: Motion capture system [2]

perimental techniques performed using the marker placement lack to bridge the gap between the muscular dynamics and human motion, and hence, the human body motion analysis is performed by mirroring the recorded human motion into a musculoskeletal simulation using tools like OpenSim [11], which is then segmented into three consecutive stages for motion analysis:

[2]https://cmasuki.org/

- Inverse Kinematics
- Inverse Dynamics
- Muscle Force Prediction

Inverse Kinematics

In inverse kinematics modeling, the best fit model is generated to match the experimental markers with the model markers for the simulated motion by minimizing the sum of weighted squared errors of the markers and coordinates throughout the motion cycle. The Inverse Kinematics (IK) toolkit, available with the OpenSim software, enables the implementation of the inverse kinematics stage. The output of this simulation stage contains generalized coordinate trajectories in the form of joint angles and translations of the motion.

Inverse Dynamics

In inverse dynamics, forces and torques, that produce the body movement, are measured at each joint, based on the kinematics information available for the motion of the model. The inverse dynamics analysis is performed using the Inverse Dynamics (ID) toolkit of OpenSim. The output of this simulation stage contains time-based joint forces and torques that produces the accelerations based on measured experimental movement and external forces acting on the model. Overall, the inverse dynamics stage focuses on solving the equation of motions for the unknown generalized forces by using the known motion of the model, where the positions, velocities, and accelerations define the motion of the model.

Muscle Force Prediction

Using the outputs of the inverse kinematics stage and the inverse dynamics stage, along with a Hill-type muscle model, the muscle forces are predicted. A Hill-type muscle model corresponds to the Hill's equations for the tetanic muscle contraction in investigations over cadavers [22].

To perform post-surgical motion analysis of the THA patients, the quality of the motion is measured based on the forces acting on the muscles during the body movement. Due to the infeasibility of the process to measure such forces, musculoskeletal

simulation is used to estimate the forces acting on the muscles during the human body movement.

2.4 Statistics : Prerequisites

2.4.1 Mean Centering

A subject dataset in an application can consist of multiple features having different measuring units and magnitude ranges. The difference in magnitude ranges will affect the significance of the features when analyzed all together. To overcome this issue, we perform mean centering on the data so that the values of features bounds around the mean zero. This process introduces a new dataset with a comparable range of magnitudes of the features that enhance the overall significance of the data.

Algorithm 1 is used to perform mean centering on every data point in a time series by subtracted by the mean of the complete series.

Algorithm 1 Mean Centering

Input: $F_1 \dots F_m \; for \; P_1 \dots P_n$
Output: *Zero Mean Centered Series*

1: **function** MeanCentering(F)
2: **for** $p \leftarrow 1$ to n **do**
3: $mean[p] = \frac{\sum_{t_o=1}^{m} F[p,t_o]}{m}$
4: **for** $t \leftarrow 1$ to m **do**
5: $F[p,t] = F[p,t] - mean[p]$
6: **end for**
7: **end for**
8: **return** F
9: **end function**

Figure 2.5 represents three time series with different range of magnitudes. After performing mean centering, the series transforms to a comparable magnitude range as shown in Figure 2.6.

2.4.2 Relative Variation Scaling

Relative Variation Scaling, algorithm 2, is a technique to normalize the data by calculating the factor of change in a time series with respect to the initial data point

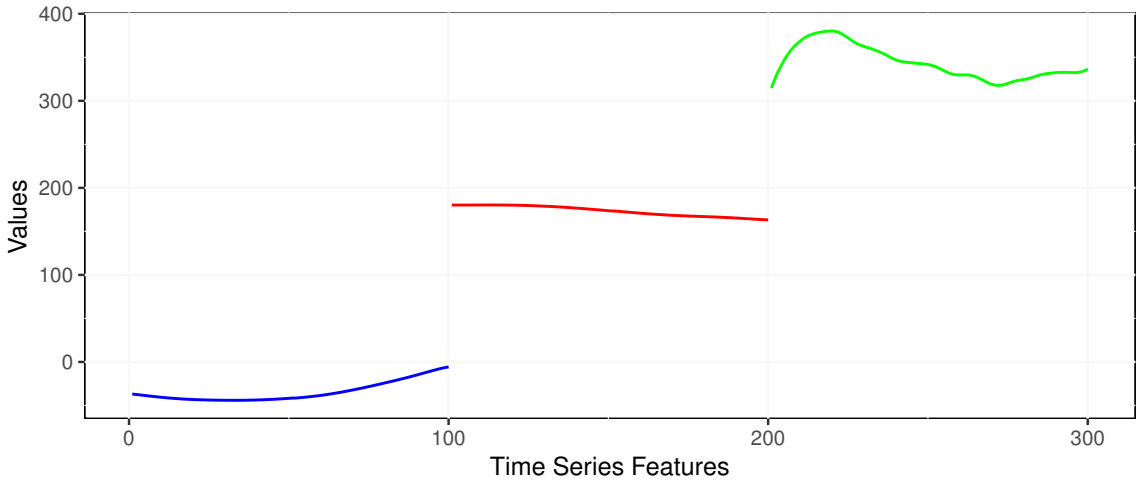


Figure 2.5: Before normalization

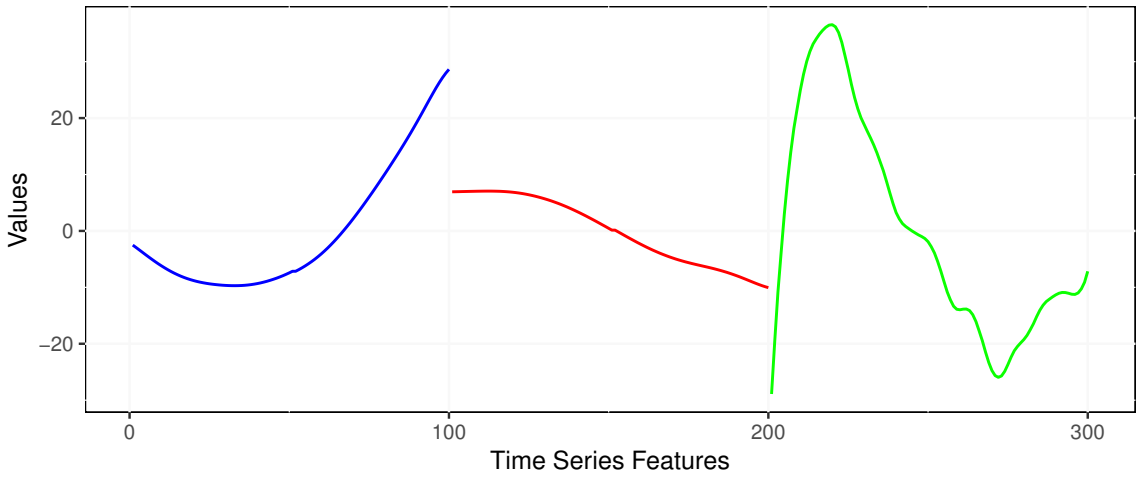


Figure 2.6: After mean centering

in the series. Applying this technique, a time series transforms to originate from value 1 and the overall magnitude range of the series gets minimized by the factor of the initial data point of the original series. By concatenating multiple time series, variability in each series can be compared.

If the time series starts with a negative number, before performing the normalization, the time series should be transformed so that every data point in the series becomes positive. This preserves the relative variation in the series, viz growth or descent.

Algorithm 2 Relative Variation Scaling

Input: $F_1 \dots F_m$ *for* $P_1 \dots P_n$
Output: *Relative Variation Scaled Series*

```
1: function RELATIVE VARIATION SCALING(F)
2:     for p ← 1 to n do
3:         for t ← 1 to m do
4:             F[p, t] = F[p, t]/F[p, 0]
5:         end for
6:     end for
7:     return F
8: end function
```

Taking the same example in Figure 2.5, after relative variation scaling, the three time series will be transformed as depicted in Figure 2.7. The overall range of concatenated series will be between the highest growth factor and the smallest descent factor.

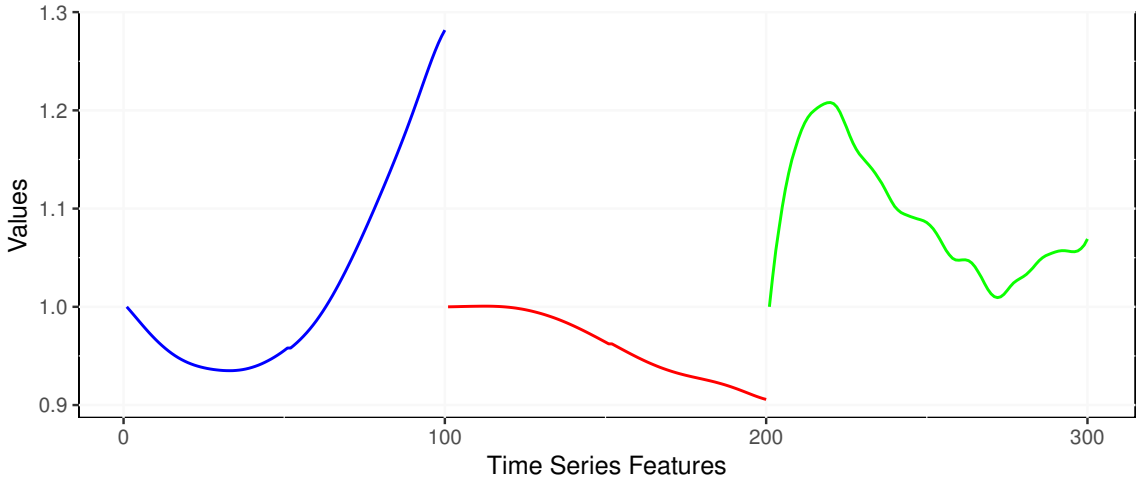


Figure 2.7: After relative variation scaling

2.5 Dimensionality Reduction

When working with multivariate time series data, the count of columns exceeds by the factor of time series intervals and increases the complexity of data by adding more dimensions to statistical regression models. To simplify the regression methods and

condense the information included in the original data to a more comprehensible form, dimensionality reduction [30] plays an important role. Principal Component Analysis (PCA) and Parallel Factor Analysis (PARAFAC) are the techniques we explore to perform dimensionality reduction. Both the methods work similarly by projecting the original data points to a set of orthogonal vectors called principal components. PCA is an efficient technique for a dataset with a shape like a 2-dimensional matrix, whereas, PARAFAC is an advanced version of PCA which applies to a more complex, multi-way dataset that has more than two dimensions. Both the techniques are used to perform dimensionality reduction on the musculoskeletal time series data. The compressed information is used to visualize the relationship between musculoskeletal inferences and the input time series data.

2.5.1 Principal Component Analysis (PCA)

Pearson coined the idea of PCA in 1901 [31]. It was also independently developed by Hotelling in 1936 [32, 33]. Phinyomark et al. provided a review of how the PCA techniques were used for gait analysis for various research questions [44] as follows:

- Differentiating between healthy and unhealthy female runners
- Predicting the response to exercise treatment for patients with patellofemoral pain and knee osteoarthritis
- Analyzing body movement differences occurring due to wearing shoes with different midsoles

Kirkwood et al. applied PCA on the gait kinematics data of elderly women suffering from knee osteoarthritis. They found the PCA technique to be effective to analyze the kinematic gait form during the gait cycle. They were able to find the most significant component which acted as the discriminatory factor between the group of females with and without osteoarthritis. This component could be used for physical therapy evaluation and treatment of the women with osteoarthritis [45].

PCA is a linear technique for dimensionality reduction of a large number of data points correlated with each other that maps the original data points to a lower dimensional linear vector called Principal Component (PC), such that the variance of the data points in the lower dimension maximizes. The variance of the original dataset is

condensed and retained cumulatively in all the PCs. The new calculated components are a transformed representation of the original data on orthogonal vectors.

When multiple PCs are calculated, the PCA algorithm orders the principal components such that retention of original variation decreases with PCs. Therefore, the first principal component will have maximum variance, and the second principal component will have the second highest variance and so on. The last principal component will account to minimum variation of the original dataset but cumulatively accounts for 100% variance of the original dataset. The PCs are the eigenvectors of a covariance matrix and hence are orthogonal to each other.

PCA is characterized by the following generative model:

$$x_{p,q} = \sum_{f}^{F} a_{p,f} b_{p,f} + \varepsilon_{p,q}$$

where $x_{p,q}$ is the measured value, $a_{p,f}$ and $b_{p,f}$ are parameters to estimate, $\varepsilon_{p,q}$ is the residual and F is the number of components extracted.

Data scaling is an essential and required step before applying PCA to the original dataset. PCA is not sensitive to the units of data points but is sensitive to the magnitude of data points. Hence, not performing scaling on the original dataset can lead to loss of information and variables with higher magnitude will become the deciding factors of the final analysis. Therefore, techniques like mean centering and relative variation scaling can be applied to the dataset before performing PCA.

Figure 2.8 illustrates the PCA with a 2-dimensional dataset by reducing the dimensions from two viz x and y, to two seperate 1-dimensional vectors called components viz PC1 and PC2. The initial random state of the components is shown by two orthogonal lines respectively. The data points are projected on the components perpendicularly, indicated by the dotted lines, and by using the length of projections, the mean squared error is computed for both components. For minimizing the mean squared error, components are adjusted spatially by translation and rotation. The final components generated by the PCA model has a minimum mean squared error of projected data points.

The two components cumulatively consist of a 100% variance of the original dataset. In Figure 2.8, PC1 has a higher variance than PC2 as depicted by an oval circum-

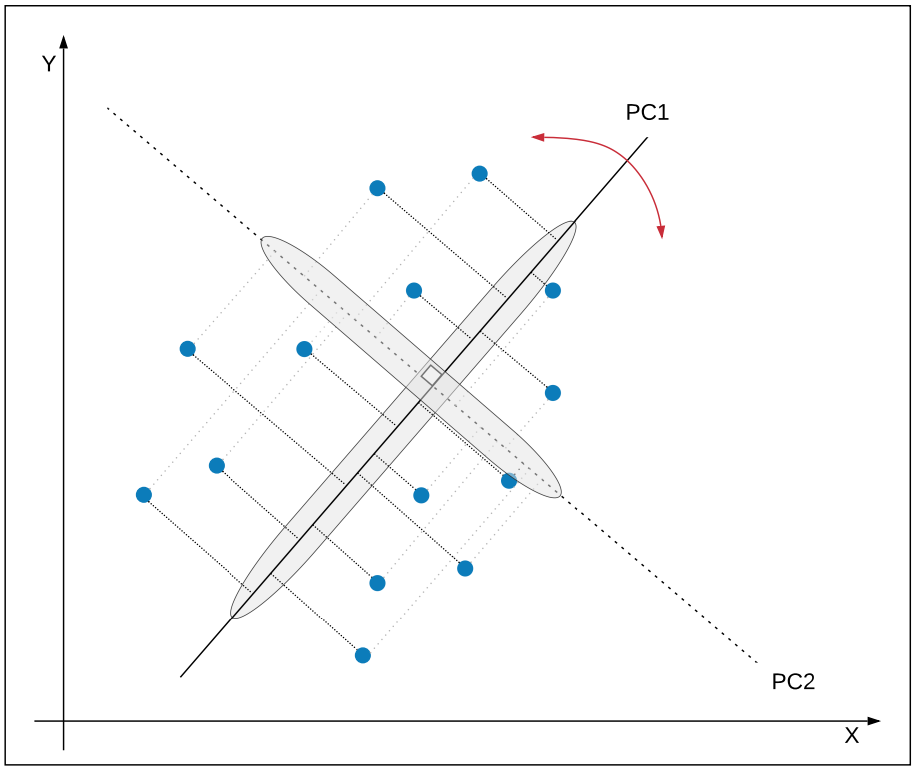


Figure 2.8: PCA, mean squared error minimization on two components

scribing the projected data points. Considering the components separately, we achieve dimensionality reduction, and each component addresses some condensed information contained the original dataset.

PCA Algorithm and Implementation

Let the dataset be

$$D = \{x^{(1)}, x^{(2)}, ..., x^{(m)}\}$$

with n dimensional inputs such that :

$$x^{(i)} \in R^{(n)}$$

1. **Data Normalization:**

   Mean is calculated across each dimension, and it is subtracted from each of the data dimension. This produces a dataset whose mean is zero. Let the mean

vector be m, such that :

$$m = \frac{1}{n}\sum_{k=1}^{n}(x^k)$$

Let the new dataset be

$$D_{norm} = \{x'^{(1)}, x'^{(2)}, ..., x'^{(m)}\}$$

where,

$$x'^{(i)} = x^{(i)} - m$$

2. **Compute Covariance Matrix:**

   Covariance matrix is computed between 2 dimensions. If the dataset has more than 2 dimensions, then multiple covariance matrices are computed.

$$\Sigma = \frac{1}{m}\sum_{i=1}^{m}(x'^{(i)})(x'^{(i)})^T$$

3. **Compute Eigenvectors and Eigenvalues from the Covariance Matrix:**

   As the covariance matrix is a square matrix, eigenvectors and eigenvalues can be calculated. The eigenvector u_1 becomes the top eigenvector for principal component 1 and subsequently u_2 becomes the second eigenvector for principal component 2. The matrix of eigenvectors can be shown as:

$$U = \begin{bmatrix} | & | & & | \\ u_1 & u_2 & ... & u_n \\ | & | & & | \end{bmatrix}$$

   Let eigenvalues be $\lambda_1, \lambda_2, ..., \lambda_n$. Here, u_1 corresponds to the largest eigenvalue λ_1, u_2 corresponds to the second largest eigenvalue λ_2 and so on. The eigenvectors are unit vectors and they help in getting useful information about the dataset. The eigenvectors forms a new basis in which the data can be represented. Using eigenvectors, length of projection of normalized data points can be calculated as following :

Let the magnitude of projection of $x'(i)$ on u_1 be $\mathsf{Mag}(x'(i), u_1)$, then:

$$Mag(x'(i), u_1) = u_1^T.x'(i)$$

Similarly, the magnitude of projection of $x'(i)$ on u_2 will be:

$$Mag(x'(i), u_2) = u_2^T.x'(i)$$

4. **Data Rotation:**

   The magnitude of projections calculated for the normalized data on eigenvectors produces a rotation of data points around the eigenvectors, and the data points are now scattered around the eigenvector. The rotated normalized data points can be calculated as follows:

$$x'_{rotated} = U^T x' = \begin{bmatrix} u_1^T x' \\ u_2^T x' \\ | \\ u_n^T x' \end{bmatrix}$$

   The eigenvectors are orthogonal to each other which satisfies the following :

$$U^T U = UU^T = I$$

   From which, it is possible to calculate back the original normalized data points as follows:

$$x' = Ux'_{rotated}$$

   as,

$$Ux'_{rotated} = UU^T x' = x'$$

5. **Dimension Reduction:**

The direction of the principal component 1 will be the direction of the rotated data on the first eigenvector. Let $\tilde{x}(i)_1$ be the transformed dataset on first principal component, Therefore:

$$\tilde{x}_1^{'(i)} = {x'_{rotated,1}}^{(i)} = u_1^T x^{'(i)} \in R$$

For reduction of n dimensions to p dimensions, where $p < n$, we can choose the first p eigenvectors. Computing them with original normalized data will convert the dataset to be projected on p dimensions:

$$\tilde{x'} = \begin{bmatrix} x'_{rotated,1} \\ x'_{rotated,2} \\ | \\ x'_{rotated,p} \\ 0 \\ | \\ 0 \end{bmatrix} \approx \begin{bmatrix} x'_{rotated,1} \\ x'_{rotated,2} \\ | \\ x'_{rotated,p} \\ x'_{rotated,p+1} \\ | \\ x'_{rotated,n} \end{bmatrix} = x'_{rotated}$$

Since, remaining $n-p$ components are zeros, we define $\tilde{x'}$ as a $p-dimensional$ vector with only the first p non-zero components.

The maximum number of principal components will be:

$$Max\_PCs = min(Num\_cols,\ Num\_rows)$$

where,
Max_PCs = Maximum number of components that can be calculated using PCA
Num_cols = Number of columns in the dataset used in PCA
Num_rows = Number of rows in the dataset used in PCA

2.5.2 Parallel Factor Analysis (PARAFAC)

PARAFAC [23] has been proved to be helpful in gait analysis of a normal person and a person with a motion discrepancy. Helwig et al. showed how PARAFAC could be used for distinguishing ankle and knee motion for the above two categories of

people [46]. It proved to be a descriptive model to interpret such shape differences with high accuracy.

Hsiao-Wecksler et al. used the PARAFAC analysis technique to distinguish symmetric and asymmetric gaits among infants, and, typical and atypical developing children [47]. They also used the method to analyze gaits of the people whose walking ability was affected due to injuries. Moreover, they also correlated the asymmetric movement with increased stress which could result in harmful effects to other body parts.

PARAFAC is a dimensionality reduction technique which is superior to PCA as it can be used with dataset having higher dimensions than two-dimensional matrices. PARAFAC simultaneously fits multiple two-way arrays or slices of a multi-way array in terms of common set of factors with different relative weights in each slice [34]. Mathematically, it is straight generalization of bi-linear model of component analysis to a tri-linear model. The PARAFAC analysis is characterized by the following generative model:

$$x_{p,q,r} = \sum_{f}^{F} a_{p,f} b_{p,f} c_{r,f} + \varepsilon_{p,q,r}$$

with an associated *sum of squares* loss:

$$\min_{\mathbf{A,B,C}} \sum_{p,q,r} \| x_{p,q,r} - \sum_{f}^{F} a_{p,f} b_{p,f} c_{r,f} \|^2$$

or:

$$Loss(a_{11}, a_{12}, ...c_{RF}) = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} (x_{p,q,r} - \sum_{f}^{F} a_{p,f} b_{p,f} c_{r,f})^2$$

where, $x_{p,q,r}$ is the measured value, $a_{p,f}$, $b_{p,f}$ and $c_{r,f}$ are the parameters to estimate, $\varepsilon_{p,q,r}$ is the residual and F is the number of factors extracted.

2.6 Statistical Regression Analysis

Regression analysis is a type of statistical modeling that estimates the relationship amongst the variables of a multivariate dataset. It examines the influence of the explanatory variables (independent/predictors) on the response variable (dependent/to be predicted) by estimating the average value of the dependent variable based on the

fixed values of the independent variables. It is significantly used for prediction and forecasting in the field of machine learning where it can determine the significant and non-significant factors in the dataset. There are two types of techniques to perform a regression:

To perform regression, data points of the response variable v/s explanatory variable are plotted to generate a best-fit regression line/curve (an equation) which depicts the correlation between the two variables. There are various types of regression models like:

- Linear Regression
- Logistic Regression
- Polynomial Regression
- Stepwise Regression

2.6.1 Logistic Regression

Logistic regression is one of the statistical regression models for predictive analysis where the dependent variable is a binary indicator variable (variable can have only two possibilities). The logistic regression method corresponds to predicting the parameters of a binomial regression based logistic model. In binomial regression, the response variable is the outcome of Bernoulli trials (success/failure or 1/0). For example, do the explanatory variables like body weight, calorie intake, exertion, and age affect the probability of having a heart attack?

Logistic regression has found many applications in the field of machine learning, medical science, and social science. For example, Boyd et al. developed a logistic regression-based method to measure the trauma and severity score of a patient with injuries [26].

Logistic regression uses a logistic function to gauge the correlation between the response variable and the explanatory variable. The logistic function/curve is a sigmoid (S-shaped) curve [27]:

$$f(x) = \frac{L}{1 + e^{-k(x-x_0)}} \tag{2.1}$$

where,

e = natural logarithm base

$x \in [-\infty, +\infty]$

x_0 = x-value of sigmoid's midpoint

L = maximum value of the curve

k = logistic growth rate or curve steepness

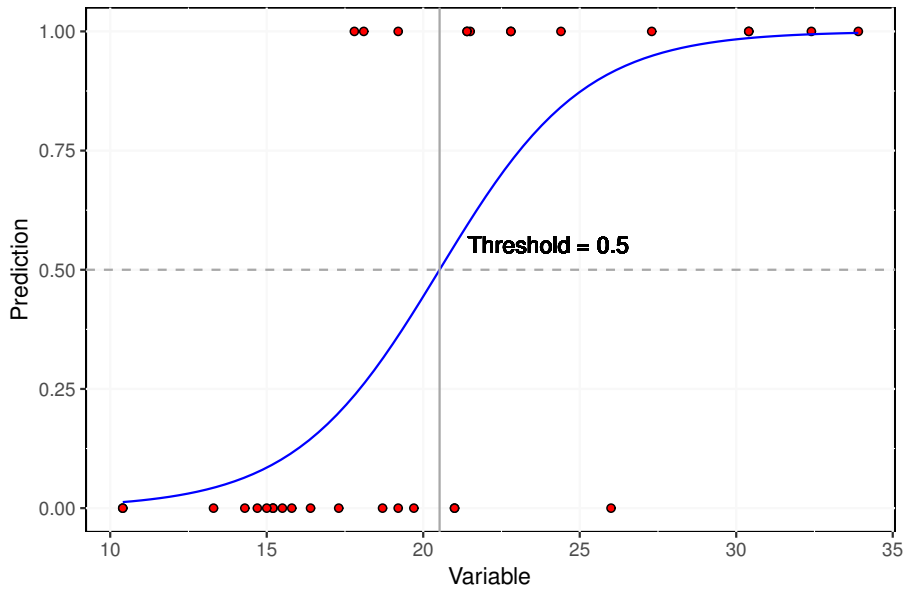


Figure 2.9: Logistic regression curve [23]

Figure 2.9 depicts the sigmoid logistic regression curve where the red dots represent the predictions corresponding to the variable value based on the threshold of 0.5. It can be seen that there are five misclassified predictions - three on the top and two at the bottom.

2.7 Neural Networks : Overview

Artificial Neural Network(ANN) is modeled into a network of interconnected artificial neurons inspired by the biological neurons of the brain. Each connection, mirroring the biological synapses of the brain, is enabled for signal transmission between the artificial neurons. The biological neurons are devised into artificial neurons in the form of mathematical functions where one or more inputs (simulating post-synaptic potentials of the biological neurons) are given to the artificial neurons which

are processed to output the results (simulating action potential of the biological neurons) depending on the input and activation (change in their internal state). Figure 2.10 represents a neuron with input-output flow via dendrites and axons respectively.

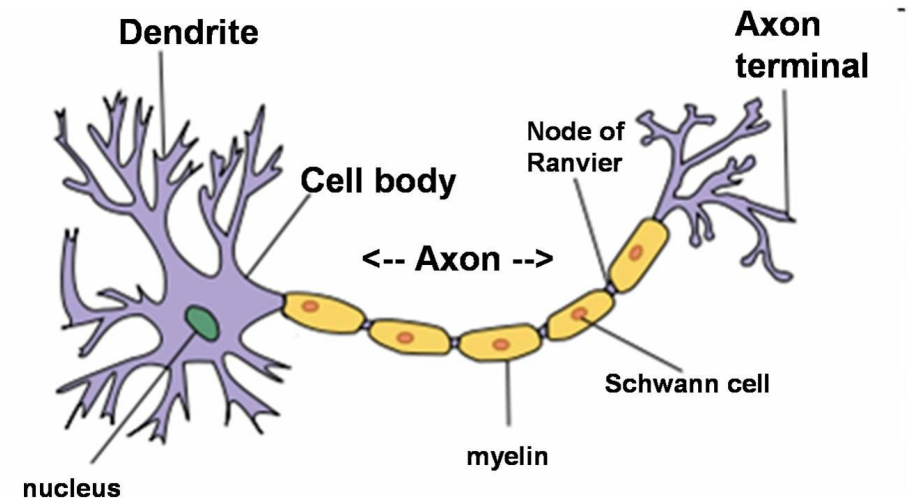


Figure 2.10: Schematic of biological neuron [38]

In 1943, McCulloch and Pitts gave an explanation of the nervous system based on the propositional logic built from the boolean characteristic of the nervous activity [5]. They used electrical circuits to model a simple neural network that functioned on the outcomes of various logical expressions that mirrored the behavior of the neurons in the brain. The model proposed by them opened the gates for exploring research in the field of applied neural networks in AI.

The ANN can be viewed as a directed, weighted graph where the network is formed by connecting the outputs of some neurons to the input of other neurons. The process of *learning* aims to modify the synaptic weights and the weights act as a memory in the network. Figure 2.11 represents the schematic of the ANN where the circular nodes represent the artificial neurons, and the arrows represent the connection between the artificial neurons (output of one neuron given as input to another neuron), simulating the synaptic connections in a biological brain. Each connection holds some weight (strength of the connection) which adjusts itself as the process of learning moves further. The artificial neurons are aggregated into multiple layers where each layer corresponds to a different transformation of the inputs. The signals travel from the

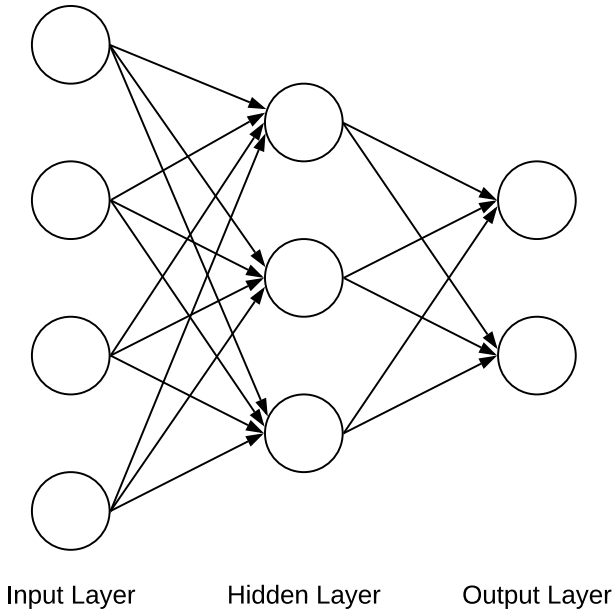


Figure 2.11: Schematic of ANN

first layer (input layer) to the last layer (output layer), traversing through multiple layers in between.

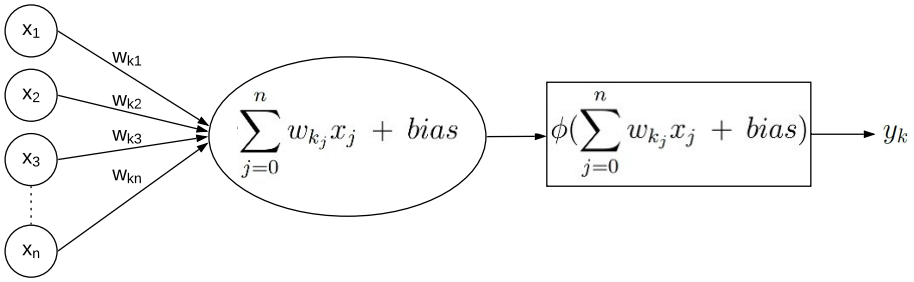


Figure 2.12: Schematic of artificial neuron [37]

Each artificial neuron can be represented as shown in Figure 2.12. Let the number of inputs given to the artificial neuron be n with input signals x_0, x_1, ..., x_n and weights w_0, w_1, ..., w_n for respective inputs. The input signals x_0, x_1, ..., x_n are the output of the predecessor neurons, and the weights are the synaptic weights of neural connections. The output of the k^{th} neuron will be calculated as follows:

$$y_k = \phi(\sum_{j=0}^{n} w_{k_j} x_j \ + \ bias)$$

where, ϕ is the activation function. The output of the artificial neuron simulates the axon of the biological neuron, the output value propagates as input to the next layer in the ANN via synaptic connection or exits as the final output of the ANN.

A multilayer ANN like one in Figure 2.11 responds to the input data with an output that corresponds to the learning of the model. It can be imitated by a mathematical function that calculates an output for a given set of input variables. Hence, a neural network is similar to an unknown function, and each layer in an ANN adds up to the composition of functions. The unknown function for the network in Figure 2.11 can be represented as follows:

$$f(X) = g(h(k(X)))$$

where X is the input vector with multiple variables or features and three function composition corresponds to the functions for each layer in the network respectively.

Learning Models

There are two major learning models used in the ANN depending on the input data:

- Supervised Learning: This learning paradigm uses example input-output pairs and designs a learning function using these pairs, where the mapping is derived from the input training data.
- Unsupervised Learning: This learning paradigm learns from the given unclassified, non-categorized and non-labeled data. Unsupervised learning identifies the common aspects present in the data and designs the learning function accordingly.

This research focuses on the supervised learning paradigm. Rosenblatt initially proposed the algorithm for pattern matching called Perceptron, a supervised learning algorithm mirroring a biological neuron [8]. Mathematically, a perceptron is a learning algorithm to learn a threshold function (a binary classifier), which maps the input vector x with weights w and bias b, to the binary output values:

$$f(x) = \begin{cases} 1 & if\ (w \cdot x + b) > 0 \\ 0 & otherwise \end{cases}$$

where,
$w \cdot x = \sum_{i=1}^{n} w_i x_i$ for n inputs.

Although the idea of collaborating the biological aspects into programming machines seemed to transfigure the world of AI, Minsky and Papert discovered the infeasibility of applied neural networks because of the limited processing power of computers and the inability of perceptrons to process the exclusive-OR circuit back in 1969 [7]. Hence research in the field of ANN became dormant until 1975 when Werbos's backpropagation algorithm resolved the exclusive-OR problem by optimizing the training of multi-layer networks [10].

Backpropagation

Backpropagation involves distributing the errors throughout the layers of the ANN in the backward direction where the errors are computed at the output. The gradient descent optimization algorithm uses the backpropagation [39] to adjust the synaptic weights of a neural network by calculating the derivative of the loss function.

The significance behind applying backpropagation is to train a multi-layered ANN in such a way that the ANN can make appropriate adjustments to weights of the connections and be able to learn to map arbitrary input to output.

2.8 Supervised Learning

Russell and Norvig gave a study of Supervised Learning where the learning function uses the example input-output pairs to learn and build an inferred function to map new inputs to outputs [6]. An optimal learning function, in this case, would be the one which can label all the new, unseen instances based on the provided input-output pairs [9].

Solving a supervised learning problem involves the determination of the following aspects:

- Type of data in the training set.
- Accumulation of the training set (collection of input-output object pairs).
- Input object representation (Input object representation significantly impacts the learning function accuracy).
- Structure and algorithm of the learning function.

- Control parameters via cross-validation.
- Accuracy on the test set.

A brief description of the working of the supervised algorithm is as follows: Consider a training set with n example input-output pairs of the form $(x_i,\ y_i)$ where $x_i \in X$, X corresponds to the input feature vector (input space) and $y_i \in Y$, Y corresponds to the output label (output space) of the i^{th} example pair. A supervised learning algorithm seeks a mapping function

$$f : X \rightarrow Y$$

The idea is to improve the approximation of this mapping function such that it can map an input data X to respective output variable Y.

2.9 ANN Variants

2.9.1 Feedforward Neural Network

The ANN described in Figure 2.11 is a feedforward network. It is the simplest form of neural network where the input data is forwarded from the input layer towards the output layer of artificial neurons to adjust the synaptic weights to perform learning. The layers other than input and output are called hidden layers which can help reduce noise in the data while propagating towards the output layer.

2.9.2 Recurrent Neural Network - Long Short Term Memory

Recurrent Neural Network(RNN) works on the principle of saving the output of a layer to use it as feedback to the same layer along with the next inputs. This helps in predicting the outcome of the layer for the input data that has subsequent information dependent on the preceding data points. The saved output from previous data maintains the persistence of information looping in the network as a short term memory.

In Figure 2.13, the first layer is formed similar to a feedforward network that is then connected to the RNN model. The RNN process starts with the arrival of output from the previous layer, and each neuron remembers the information from the last time-step to be used as feedback for the current time-step. Similar to feedforward,

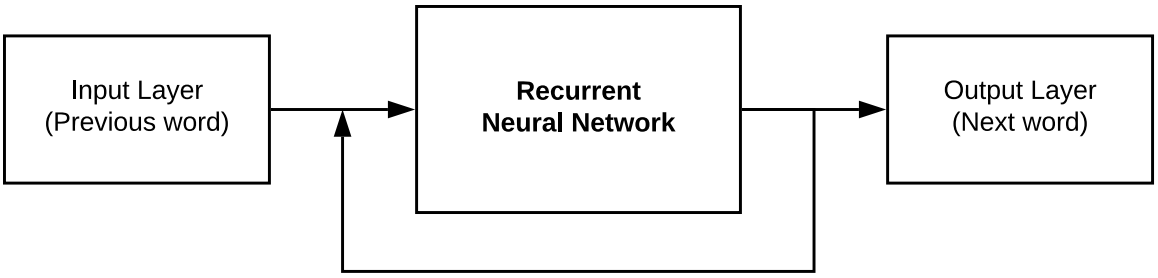


Figure 2.13: Recurrent neural network

RNN uses loss and optimization functions to reduce the error rate and gradually tunes the network to make correct predictions with the help of back-propagation. Figure 2.13 shows a basic network for prediction of the next word in a sentence.

Long Short Term Memory (LSTM) is a special kind of RNN that is competent in learning long term dependencies. Hochreiter and Schimidhuber first introduced it in 1997 [21] which further got improved by researchers. An RNN is formed by a chain of repeating modules of the LSTM units.

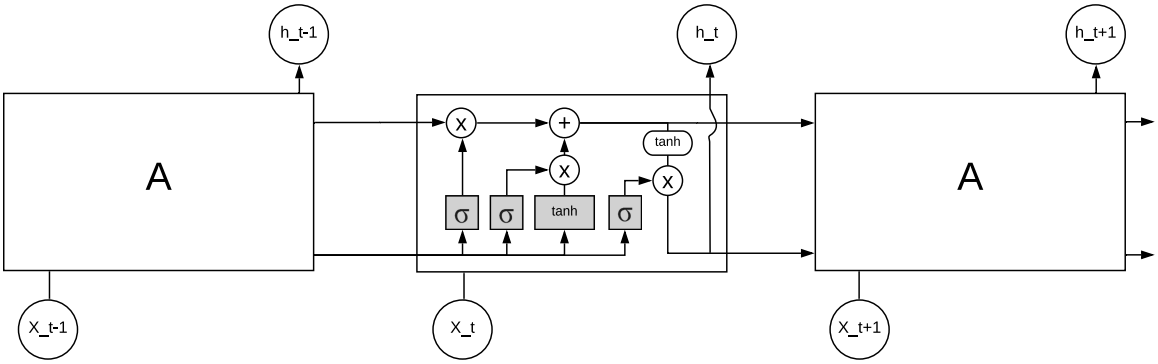


Figure 2.14: Long Short-Term Memory

Figure 2.14 is a chain of three LSTM units each having the same structure as the middle one. The grey boxes are neural network layers, circles with x and + are pointwise operators for vector multiplication and addition respectively, and the arrows are vector transfers. Each arrow carries output from one node to another as an input vector. The line running at the top carries a cell state throughout the chain which undergoes minor operations. LSTM can add or remove information in the cell state regulated by a gate which is composed of the output of the sigmoid neural network

layer and pointwise multiplication operator. The LSTM has three of these gates to control information in the cell state.

The first gate decides which information needs to be removed from the cell state, which is called as the "forgot gate layer". The next gate controls the new information that will be added in the cell state, which is called as the "input gate layer". The third gate is the "output gate layer" which decides information that should be the output from the cell. Tanh function is used to convert the vectors to a range of numbers between -1 and 1.

2.9.3 Convolutional Neural Network

Convolutional Neural Networks (CNN) are an advanced version of feedforward neural networks. CNN have proven efficient in the areas such as image recognition, classification and signal processing. CNN have had successful applications in face recognition, object identifications and combining many applications, CNN has been powering the robotics significantly. CNN has become an important tool for machine learning researchers and practitioners due to its wide variety of capabilities.

Yann LeCun developed one of the very first CNN [24] and was named LeNet5 after many previous successful iterations since 1988. It was mainly used for character recognition for applications like reading zip codes, digits, and so on.

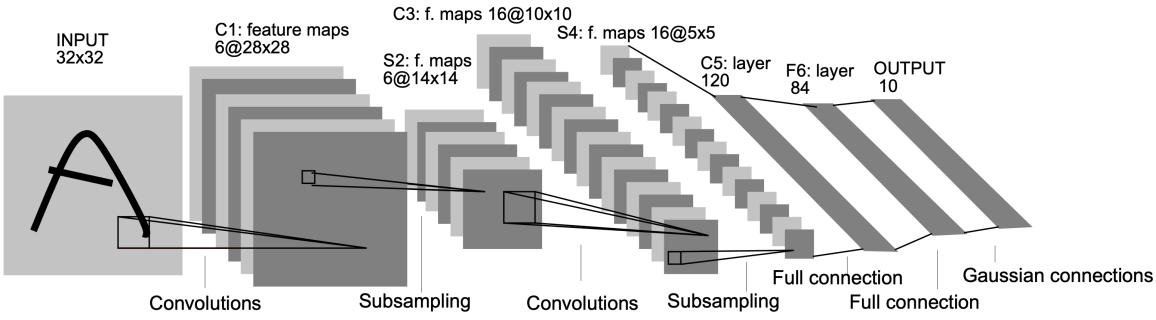


Figure 2.15: LeNet5 [24]

Figure 2.15 depicts the CNN developed by Yann LeCun [24] which illustrates a character recognition application using a handwritten letter "A". CNN layers are the composition of 3 different layers namely convolution, pooling, and non-linearity. The convolutions are responsible for extracting spatial features of the input image, pooling

extracts average of the pool of a particular size from the output of convolutions and non-linearity is the output formed by non-linear activation functions like tanh or sigmoid. The result is further flattened into a 1-dimensional vector to be forwarded to a feedforward part of the network which is the final classifier.

During 1988, the computation power we had was inadequate for heavy computational models of neural networks. With the advent of powerful CPUs, neural network architectures support better applications with faster computations and better results. In 2012, Alex Krizhevsky invented a deeper and wider version of LeNet called AlexNet [25].

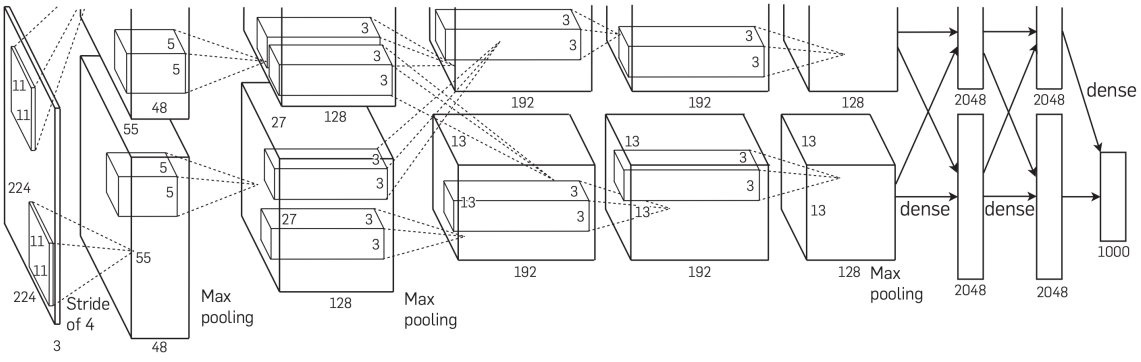


Figure 2.16: AlexNet [25]

Figure 2.16 illustrates the AlexNet CNN which is a more extensive neural network than LeNet that could be used to learn much more complex figures than alphabets and digits. The work in AlexNet contributed in use of rectified linear unit (ReLu) as non-linearities, use of dropout techniques to discard some neurons during training, back-propagation that could be a way to circumvent over-fitting the model, and use of max-pooling instead of average pooling which selects the most significant pixel from the output of convolution parts. The "Deep Neural Network" term was coined after the larger network was used for more complex learning with 10x faster training speeds by deploying Graphics Processing Units (GPUs).

The parts of CNN are defined as follows:

1. **Kernel:** It refers to a restricted subarea of input receptive field which is the output of the previous layer. It is a two-dimensional $n \times n$ matrix, typically square-shaped, which is called the size of a kernel, for example, 5×5. The kernel moves around the input space to all possible locations projecting smaller

input shapes to the pooling layer. A kernel is also called as a filter or a neuron. It is called neuron because the subarea of the receptive field acts as a specific input vector for a neuron in the convolutional layer.

2. **Stride:** The kernel slides over the input receptive field, which is called convolving, and the number of pixels the kernel shifts is called a stride. The stride governs the size of the output of a convolutional layer which can help reduce spatial dimension and with different number of shifts, it can provide a better sectional separation to the kernels.

2.9.4 Undercomplete Autoencoder

This type of feedforward neural network works on the principle of supervised learning, Section 2.8, where the input data and output data is provided, and the neural network is trained to yield the same results. While training, the input data and output data provided to the network are the same, that is, the network tries to reconstruct the provided input data. The structure of an undercomplete autoencoder, as shown in Figure 2.17, is such that the hidden layers have a lesser number of neurons than the input and output layers. In the first part of the network, called encoder, subsequent layers have a decreasing number of neurons, and in the second part of the network, called decoder, subsequent layers have an increasing number of neurons and have the exact opposite architecture as that of an encoder. Therefore, the middle layer has the least number of neurons that can yield the most significant information contained in the original input in an encoded format, that can be used to reconstruct the original input with reduced noise.

The result of undercomplete autoencoder is fetched from the middle layer, and that is why undercomplete autoencoder is a type of dimensionality reduction technique that uses neural networks. For example, if the input and output layer has total 200 neurons, which should be the size of an input vector, and the middle layer has a total of 20 neurons, the output generated from the middle layer will have a total of 20 data points per sample. That is how the undercomplete autoencoder helps reduce the dimension by 10x.

Undercomplete autoencoder is inspired by the idea of a statistical dimensionality reduction technique called PCA 2.5.1 where the reduced dimensions are known as

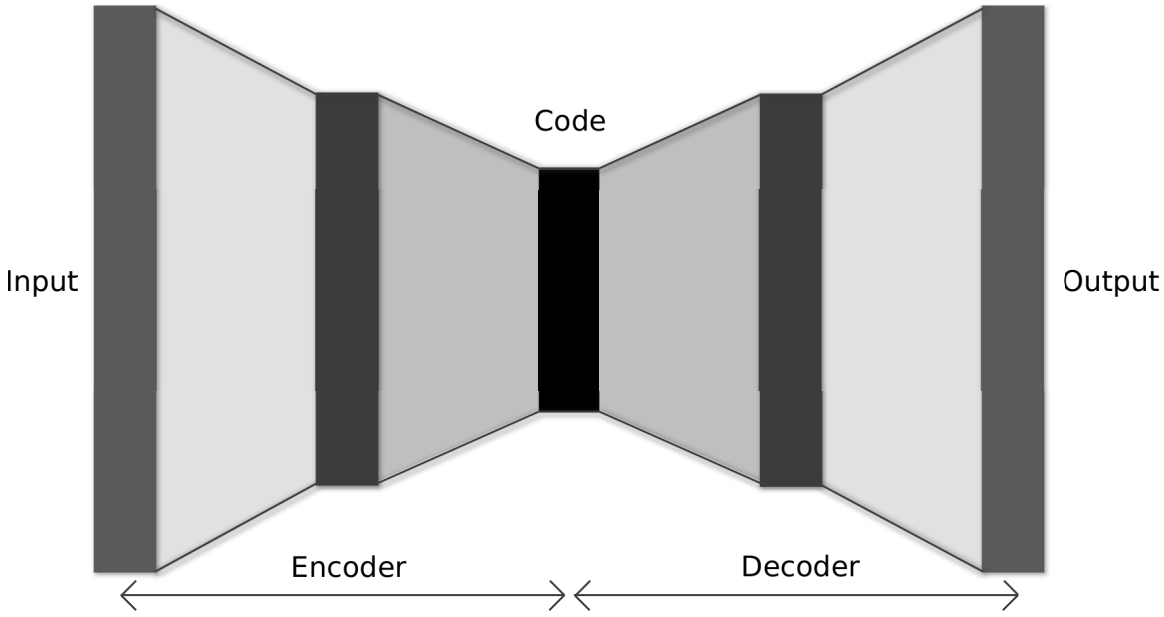


Figure 2.17: Undercomplete autoencoder

components. Using this neural network encoding technique, the most important information about the input dataset can be concentrated in the middle layer which can be further used for classification using neural networks with improved accuracy and minimized training duration.

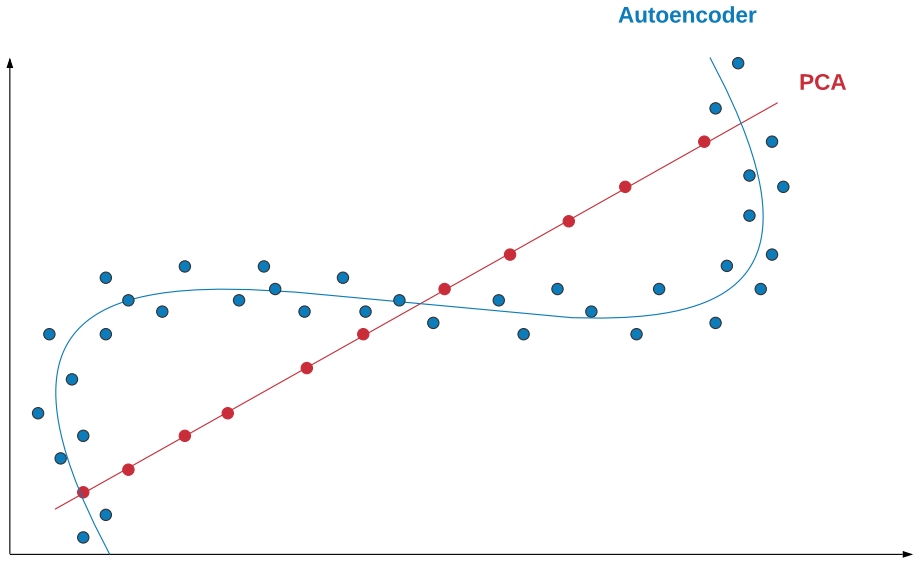


Figure 2.18: Autoencoder contrast with PCA 2.5.1

In contrast with PCA 2.5.1, neural networks can learn non-linearity and hence, autoencoders are more powerful than the linear dimensionality reduction technique PCA 2.5.1. The components extracted from a PCA model are orthogonal and linear, but the components extracted from an autoencoder can be non-linear in nature and can describe the original dataset with better information than that of a PCA model. This is illustrated in Figure 2.18.

Artificial Neural Networks have been helpful in cancer diagnosis for various cancers like lung cancer, prostate cancer, and breast cancer. Floyd et al. developed a neural network model to predict breast cancer from the mammography of 260 patients. They had these biopsies analyzed by radiologists as well as fed them to the ANN model and found that the network predicted cancer with more accuracy than the radiologist [43].

Danaee and et al. devised a deep learning approach for detecting breast cancer by identifying the critical genes [42]. They used the autoencoder to extract relevant features from the high dimensional and complex gene expression profiles. They used supervised classification models to evaluate the performance of the extracted features to use for cancer detection.

Figure 2.19 depicts the ANN approach used over the gene representation dataset for breast cancer detection. The original dataset was divided into training and testing datasets. The training dataset was used in autoencoder for noise reduction and fetching the most relevant information in the dataset using dimensionality reduction. The results of this process were then used for the classification in breast cancer detection.

2.10 Activation functions

In ANN, the activation function defines the output of a neuron for a given set of inputs. This output of a neuron acts as an input to a neuron in the next layer, and successively the activation function makes transformations to the neuron's output until a final desired output is found for the original problem. The activation function maps the calculated value of a neuron to the desired range of numbers depending on the activation function used. The transformations can be linear, that is binary: ON and OFF, or non-linear, that is a continuous range of numbers, for example, numbers between 0 to 1. Generally, non-linear activation functions produce better learning for non-trivial problems with a comparatively smaller size of the network and a lot less

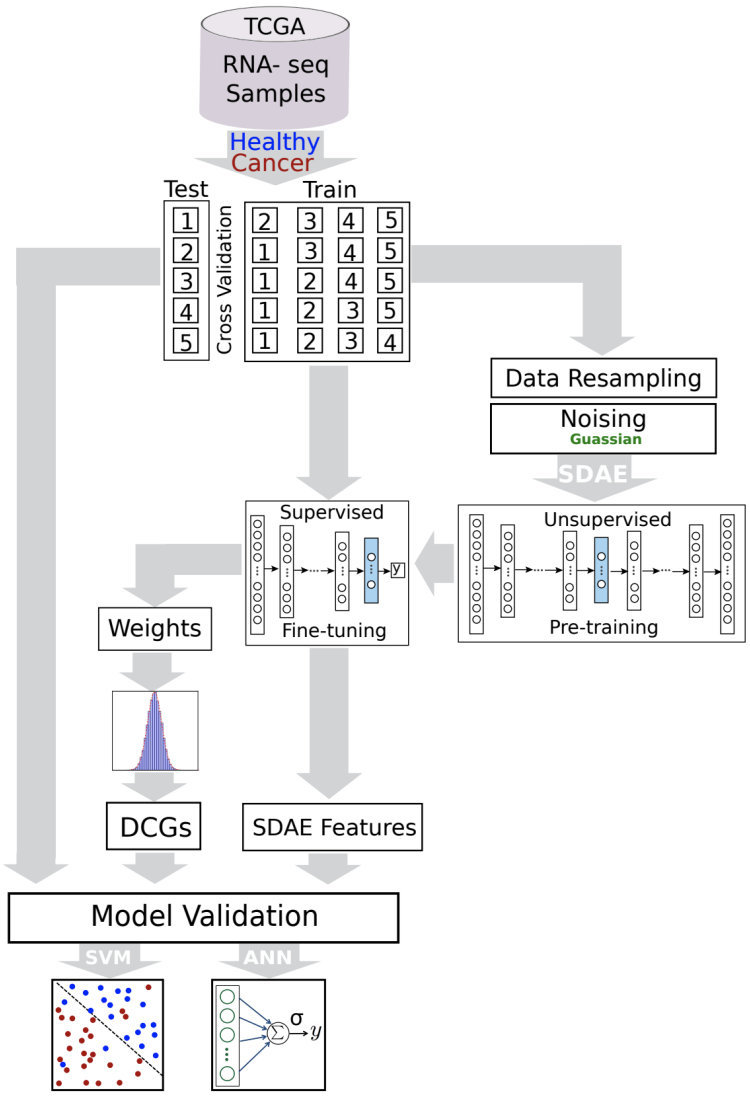


Figure 2.19: ANN approach by Danaee et. al for cancer detection [42]

computation. With non-linear activation functions, it makes it easy for the model to generalize with a variety of data and to differentiate between the output [35]. The magnitude of the output of a neuron can be considered as to how active the neuron is for a particular input. In the case of linear activation function, it comes down to whether the neuron is firing or not.

For this research, the following activation functions are used for experimental analysis.

2.10.1 Non-Linear Activation Functions

The following terminologies can be used to understand and differentiate the non-linear activation functions:

1. **Differentiable:** A function that has a change in y-axis with respect to change in the x-axis, more precisely, the slope of the curve changes w.r.t change in the x-axis, falls in the category of differentiable activation functions [36].
   Let $\phi(x)$ be the activation function and I be the input dataset, then

   $$\frac{\delta\phi(x)}{\delta x} \neq 0 \ , \ \forall \ x : \{x \in I\}$$

   The derivatives are helpful to understand about the direction and magnitude of change in learning curve of a neural network.
2. **Monotonic:** A function is monotonic when it is either entirely non-decreasing or non-increasing:

   $$\frac{\delta\phi(x)}{\delta x} \geq 0 \ , \ \forall \ x : \{x \in I\}$$

   or

   $$\frac{\delta\phi(x)}{\delta x} \leq 0 \ , \ \forall \ x : \{x \in I\}$$

2.10.2 Sigmoid

$$\phi(x) = \frac{1}{1 + e^{-x}}$$

Sigmoid, a type of logistic activation function, is non-linear that has a range between [0, 1]. Therefore, it is useful for the models that predict the probability as output, generally for multi-class classification problems. The probabilities received from sigmoid might not sum up to 1. Therefore, it is used mostly as the activation function of input and hidden layers rather than the output layer. It a **Differentiable** function and the slope of the sigmoid curve can be found at any two points. It is also **Monotonic** in nature but the differential of the curve at any two points is not monotonic.

One limitation of the sigmoid function is the possibility of neural networks getting stuck during the training. The reason for these limitations derives from the range of

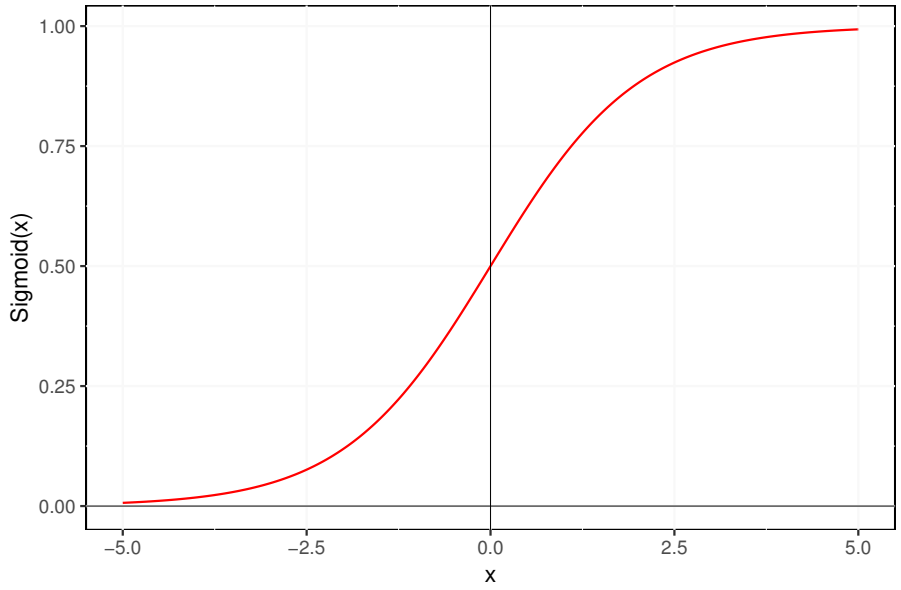


Figure 2.20: Logistic sigmoid function

the output of the sigmoid function, which is $[0,\ 1]$. In the case of strongly-negative inputs, the output of the function will be close to zero that consistently keeps the neurons in the inactive state, due to that it inhibits the neural network learning and gets stuck in its current state.

2.10.3 Hyperbolic Tangent (tanh)

$$\phi(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$

Tanh function is similar to sigmoid, being a scaled version of it.

$$\tanh(x) = 2sigmoid(2x) - 1$$

This variation of the logistic activation function is also non-linear, having a range between $[-1,\ 1]$. It a **Differentiable** function and the slope of the tanh curve can be found at any two points. It is also **Monotonic** in nature and derivatives are steeper as compared to sigmoid. As the output range of tanh covers negative and positive values, it eliminates the limitation of the sigmoid function. When the inputs are strongly negative, the output of tanh will be close to -1 and when the inputs are

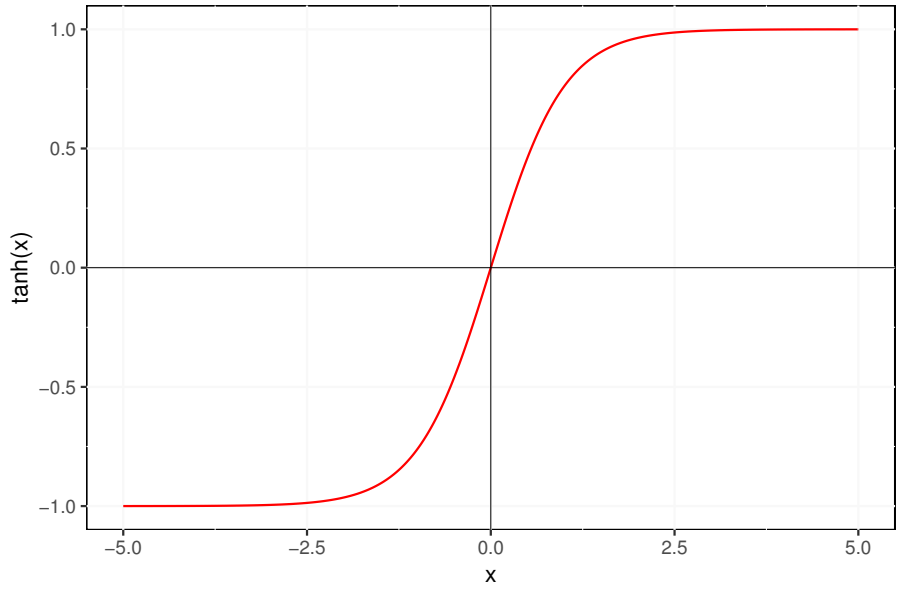


Figure 2.21: Hyperbolic tangent function

strongly positive, the output will be close to +1. Similarly, if the inputs are close to zero, the output of tanh will reflect a similar output being close to zero. Therefore, it covers the whole range of input values and makes the neural network less likely to get stuck during training.

2.10.4 Rectified Linear Unit (ReLu)

$$\phi(x) = \max(0, x)$$

ReLu, also known as the ramp function, only covers the positive part of the input and eliminates the negative by substituting it with zero. Its output range is $[0, \infty]$. When the output of ReLu is zero, the neurons are not firing and that affects the learning of a feedforward neural network. Relu is non-linear when both positive and negative axes are considered.

In neural networks, when the emphasis is on the positive input values, it can produce sparse activation of network's units. Approximately, 50% of units in the network will not have any activation, and the other neurons will be firing an output

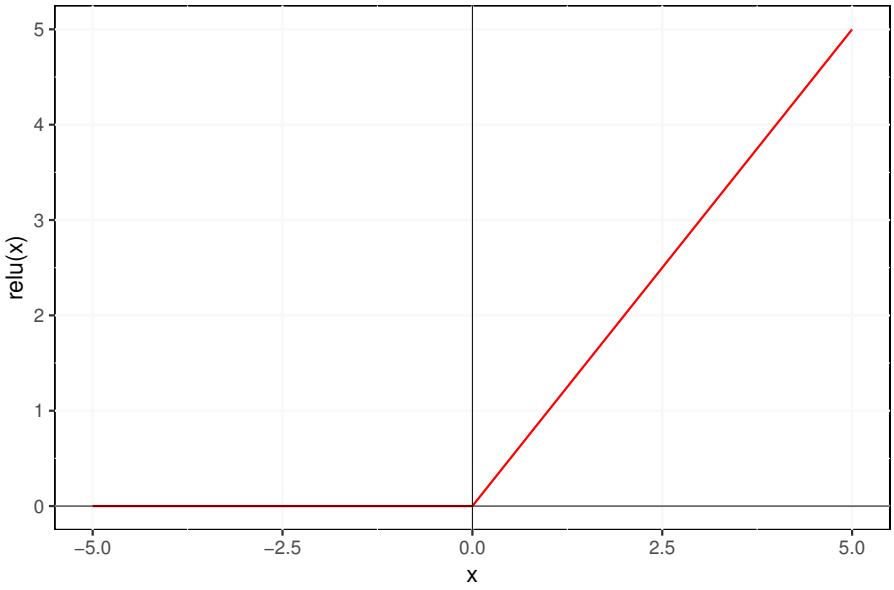


Figure 2.22: Rectified linear unit

with an infinite numerical range. About, half of the network remains in an idle state during the learning process that significantly lowers the computation.

2.10.5 Softmax

$$\phi(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}} \quad \forall\, j : \{j \in \ \{1, ..., J\}\}$$

Also a generalization of logistic function, softmax function takes an unnormalized vector and converts it into a probability distribution. Similar to sigmoid, the range of the output of softmax is also between $[0,\ 1]$. Irrespective of the sign and magnitude of input vector elements, softmax function normalizes the input vector such that:

$$\sum_i x_i = 1$$

Since the sum of all predictions equals to 1, softmax is a suitable option for activation function to be used for multi-class classification problems and should be used in the output layer of a neural network that predicts a class of an input set. The predictions

for all the classes are compared, and the class with the highest probability becomes the predicted class for that input set.

Chapter 3

System Description

3.1 Data Collection

The Human Dynamics Laboratory at the University of Denver collected the data for the "sit-to-stand" motion of 48 patients who had undergone the THA (Total Hip Arthroplasty). Patients were eligible to participate in the research if their age was between 45 and 80 years, with body mass index less than 40 kg/m^2. Apart from these constraints, the participating patient should be void of history with diabetes, hypertension, orthopedic pathology, and neurological disorders. The 12 weeks long post-operative surgical analysis (performed after ten weeks post surgery), approved by the Colorado Multiple Institutional Board, was performed with the patient's written consent [3].

Thirty-two reflective markers were placed on each patient to capture the "sit-to-stand" motion anatomically, from a chair of height 43 cm, into a simulated musculoskeletal model. For the "sit-to-stand" task, the motion cycle under analysis started from the static sitting position on the chair. The markers were placed on the torso, pelvis, thigh, shank, and foot. The body motion was performed on a Bertec force platform [12] embedded in the floor which collected the data of ground reaction force at 2000 Hz while the Vicon motion capture system [13] consisting of 8 cameras recorded the data of marker placements for one motion cycle at 100 Hz.

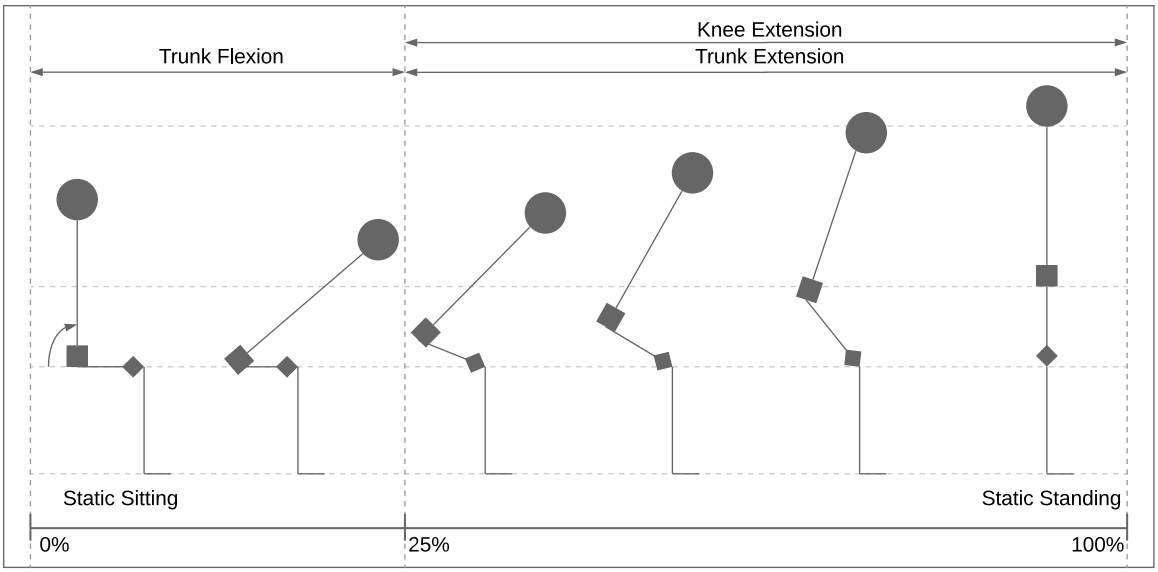


Figure 3.1: Sit To Stand Motion Cycle

The isometric strength of the hip flexors, extensors, abductors, and knee flexors and extensors, of the affected limb, were gauged with the help of electromechanical dynamometer [14] connected to a Biopac data acquisition system [15].

3.2 Musculoskeletal Modelling

The muscle strength of each patient's lower extremity is quantified with the help of a musculoskeletal model which is inspired by the hip musculature system. A combination of Gait2932 and lower limb extremity model available with OpenSim software was used for the examination and analysis of the three different stages of musculoskeletal modeling:

- Inverse Kinematics
- Inverse Dynamics
- Muscle Force Predictions

Figure 3.2 depicts the calculation of the variables comprising the three stages of musculoskeletal modeling.

The marker-base motion capture system is used for measuring inverse kinematics, which is a dataset consisting of joint angle time series The measured inverse kinematics and estimated body segment parameters (mass, the center of mass and moment of

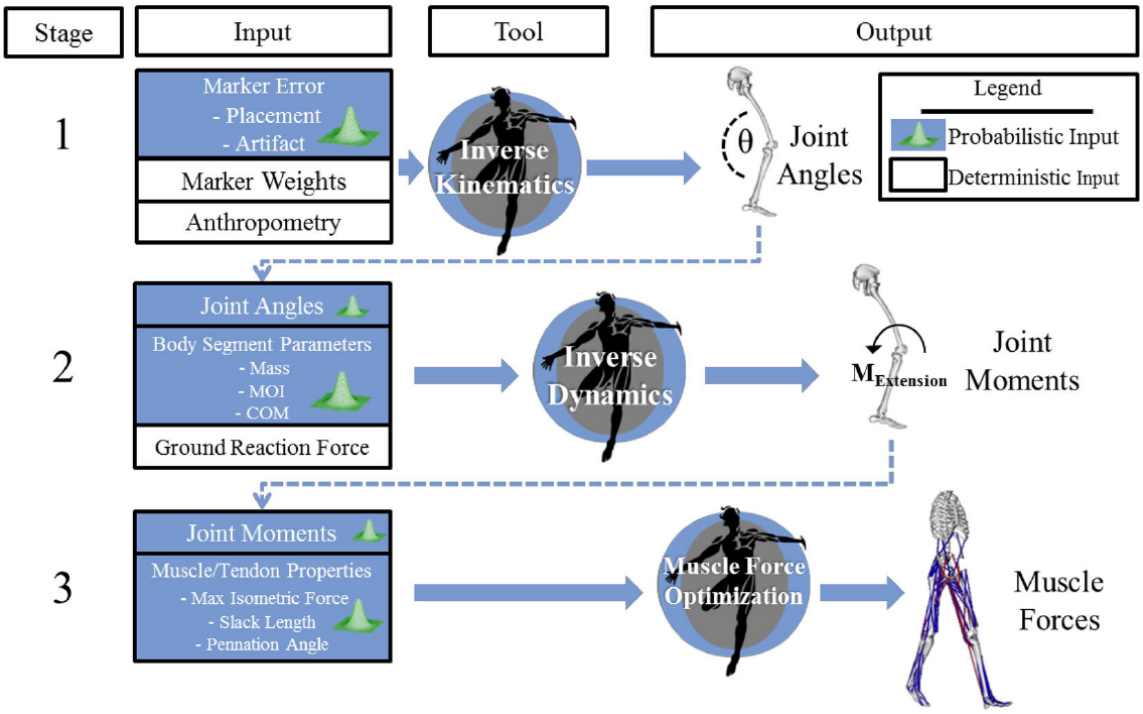


Figure 3.2: Three stages of OpenSim motion analysis [40]

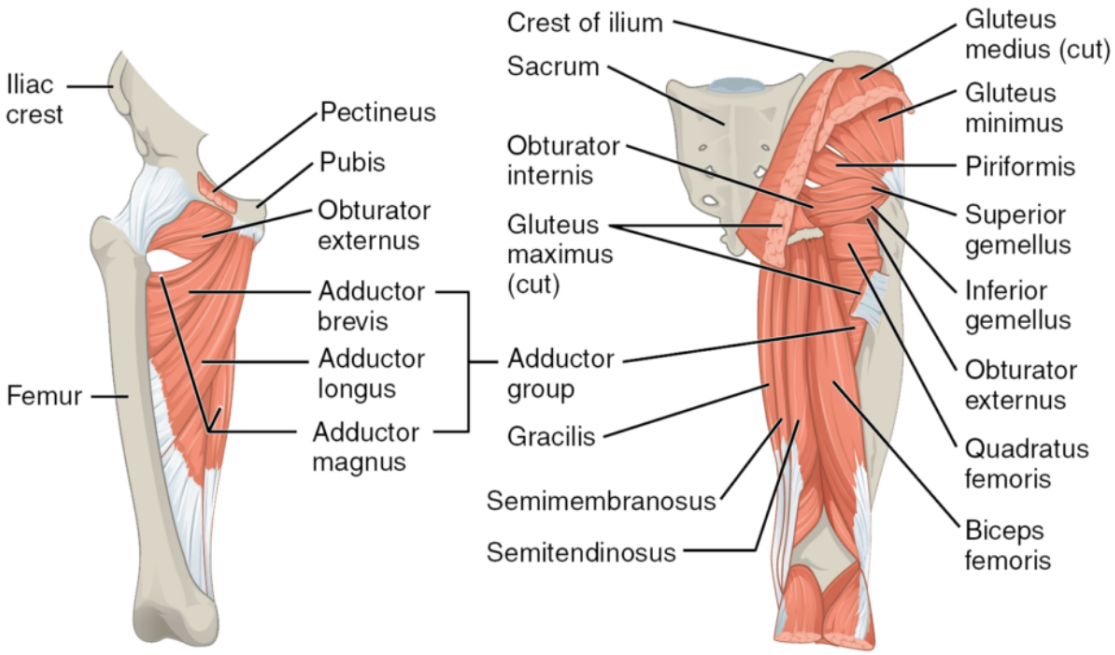


Figure 3.3: Hip Musculature 1 [3]

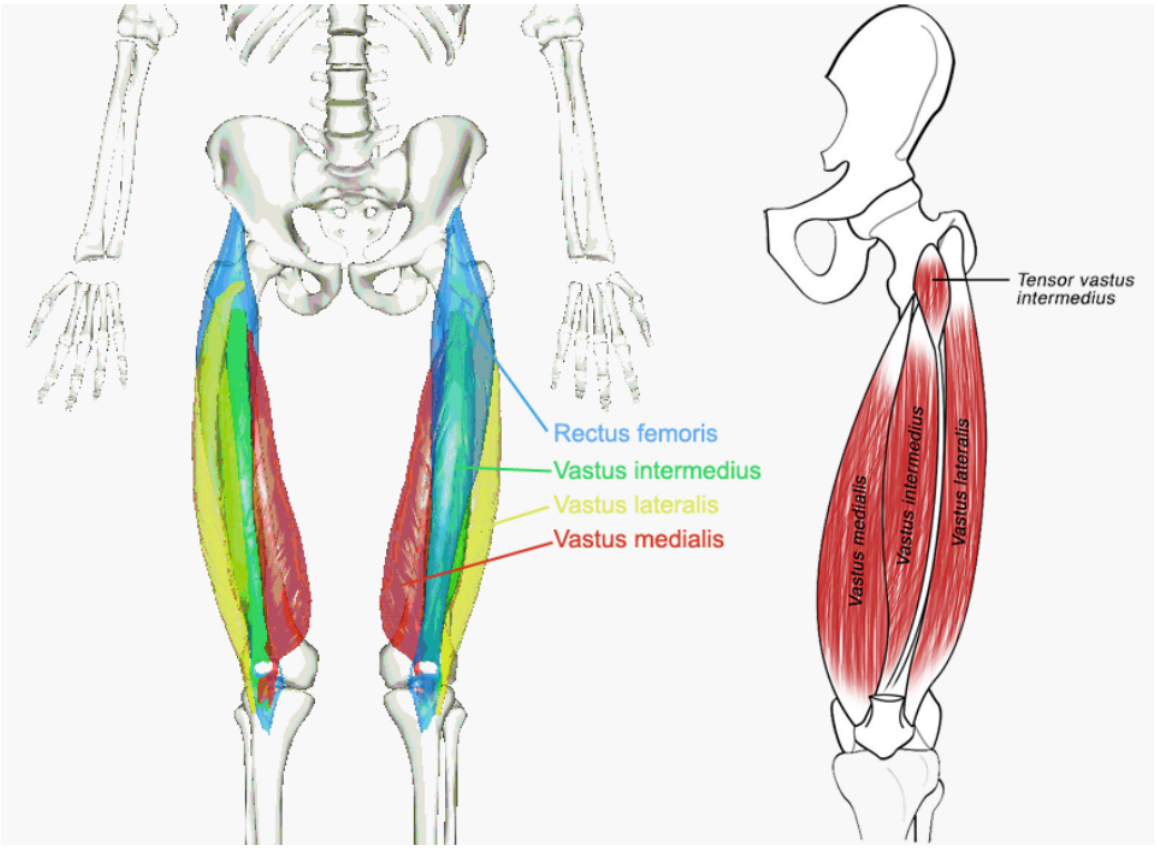


Figure 3.4: Quadriceps femoris muscles [3]

inertia) are used to calculate the inverse dynamics, which is a dataset consisting of joint forces during a motion cycle. The third stage of muscle force predictions utilizes the data of inverse kinematics, inverse dynamics and measured isometric strength of skeletal joints along with the data from the Hill type muscle model to generate muscle force predictions.

The musculoskeletal models used are 3-D computer models for the human musculoskeletal system with 23 degrees of freedom. The model comprises of 92 musculotendon actuators which represent various muscles present in the lower extremities. The analysis has been centered around the following muscles from the hip musculature as shown in Figure 3.3 and 3.4:

- Gluetus maximums
- Gluetus minimus

[3]https://lumenlearning.com/

- Gluetus medius
- Semimembraneous
- Semitendinosus
- Tensor faciae latae
- Quadriceps femoris muscles
  - Rectus femoris
  - Vastus intermedius
  - Vastus lateralis
  - Vastus medialis

To mirror a patient's musculoskeletal structure on the virtual musculoskeletal model, scaling was performed to match the relative distances between the experimental markers on the patient's body. Body segment dimensions, mass properties, muscle actuators, and wrapping objects of the virtual model were also scaled according to the actual values of the patient's body segment parameters. Baseline simulation was used to measure kinematics and ground reaction forces for each patient's musculoskeletal model which helps to estimate the joint contact forces and muscle forces.

3.3 Data Definition

The data collection and simulation process created a numerical dataset of 48 patients that is composed of time-series features like joint angles, joint forces and muscle forces for one cycle of "sit-to-stand" motion, where each feature consists of values for 100 time units. The features are categorized under inverse kinematics, inverse dynamics, muscle forces, and body segment parameters as follows:

3.3.1 Inverse kinematics

The following features (unit: degrees) are related to the pelvis.

1. Pelvis tilt
2. Pelvis list
3. Pelvis rotation

The following joint angles (unit: degrees) are simulated for the right and left sides of the lower extremity.

1. Hip flexion
2. Hip adduction
3. Hip rotation
4. Knee flexion
5. Ankle angle

The features (unit: degrees) related to the rotation of the abdominal segment are as follows:

1. Lumbar bending
2. Lumbar rotation

The following features (unit: meters) are related to the translation and extension of the abdominal segment.

1. Pelvis translation (3-dimensional)
2. Lumbar extension

3.3.2 Inverse dynamics

The following joint forces (unit: newtons) are simulated for right and left side of the lower extremity.

1. Hip joint force (3-dimensional)
2. Hip joint force magnitude
3. Knee joint force (3-dimensional)
4. Knee joint force magnitude

3.3.3 Muscle forces

The following muscle forces (unit: newtons) are simulated for right and left side of the lower extremity.

1. Semimembranosus force
2. Semitendinosus force
3. Biceps femoris-long head force
4. Biceps femoris-short head force
5. Gluteus maximus force
6. Rectus femoris force

7. Vastus medialis force
8. Vastus intermedius force
9. Vastus lateralis force
10. Medial gastrocnemius force
11. Lateral gastrocnemius force
12. Soleus force

3.3.4 Body segment parameters

The following features are recorded in the motion capture system.

1. Mass (unit: kilo-grams)
2. Ground reaction force (right and left leg) (3-dimensional), (unit: newtons)
3. Center of pressure (right and left feet) (2-dimensional), (unit: meters)
4. Moment (right and left leg), (unit: newton-meters)

Detailed information about the features can be found in Appendix A.3.

The original dataset is provided in a file with 7502 columns and 48 rows that consists of time-series musculoskeletal information of 48 patients for one "sit-to-stand" motion cycle. There are a total of 75 time series features for a motion cycle, each divided into 100 time units and two singular value features, weight and height of patients.

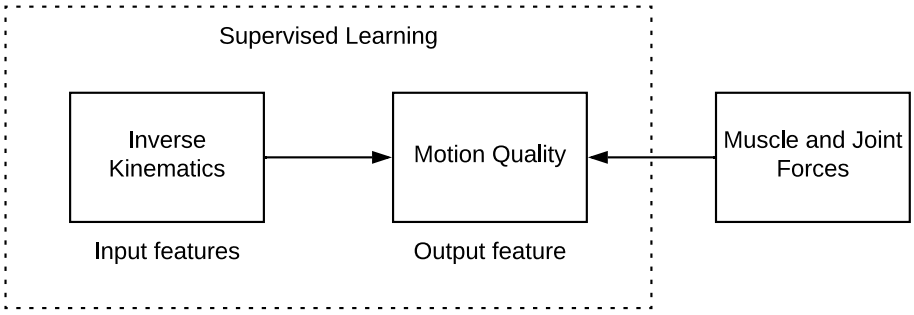


Figure 3.5: Supervised learning using the dataset

For this research, we perform supervised learning for motion quality prediction using Inverse kinematics as the input feature and motion quality, quantified using muscle forces and joint forces, as the output feature.

3.4 Data Normalization

Weight Normalization

Features with units in newtons and newton-meters have a direct correlation with the weight of the patient. Hence, it is essential to normalize the features such that the values are independent of the weight of the patients and this makes it possible to compare features like muscle forces or joint forces of different patients with each other. The final values that will be received by weight normalization will be a factor of the weight of the patient. So, the forces acting on the muscles and joints will be proportional to body weight.

Let F be a matrix of a feature with unit either newtons or newton-meters for all 48 patients. Let M be a vector of patients' mass. The following approach is used to perform weight normalization:

Algorithm 3 Weight Normalization

Input: $F_1 \dots F_{100}$ *for* $P_1 \dots P_{48}$, $M_1 \dots M_{48}$
Output: *Weight Normalized Series for each Patient*

```
1: function NormalizeByWeight(F, M)
2:     for p ← 1 to 48 do
3:         for t ← 1 to 100 do
4:             F[p, t] = F[p, t]/(M[p] ∗ 9.8)
5:         end for
6:     end for
7:     return F
8: end function
```

Mean Centering

As discussed in Section 2.4.1, mean centering is an important step to be performed when a combination of variables is formed by selecting relevant features from different sets of units like newton-meters, newtons, meters, and degrees as there can be a difference in the magnitude ranges which can affect regression models and neural network predictions. Even if the features have the same unit, the magnitude can differ based on the application on different subjects. After applying this process, the overall maximum and minimum range can be compressed significantly, and every variable will have an increased significance of its properties.

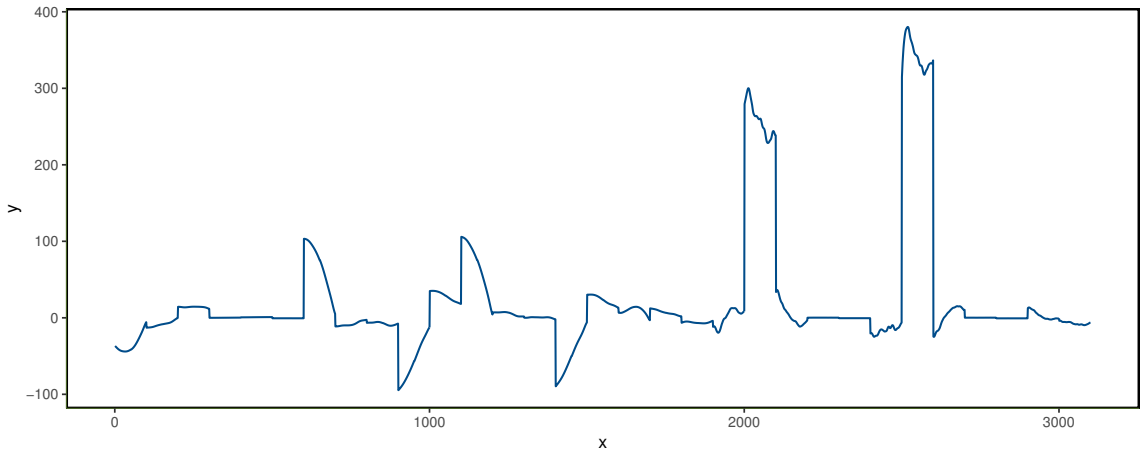


Figure 3.6: Original input sequence for patient 1

Figure 3.6 represents a combination of 31 time-series consisting of original inverse kinematics and body segment parameters, stacked one after the other in the form of a one-dimensional vector. The plot shows variable magnitudes having units in newtons, degrees, and meters. The figure suggests that there is a scope of normalization as peaks with higher magnitudes are present along with lower magnitude values.

Mean centering if performed for all the features of 48 patients using Algorithm 1 where each data point in a series is subtracted by the mean value of that series.

After performing weight normalization on some features and then mean centering on all the features, the original sequence of 31 features, as shown in Figure 3.6, transforms to a sequence of new values as shown in Figure 3.7.

Relative Variation Scaling

As discussed in Section 2.4.2 for performing relative variation scaling, every data point in each time-series is divided by the value of the first data point of the series. Applying this normalization approach transforms the dataset to a new data consisting of relative growth factors in each time-series.

After performing the normalization using Algorithm 2 on the time-series as shown in Figure 3.6 transforms the sequence to a new sequence as shown in Figure 3.8.

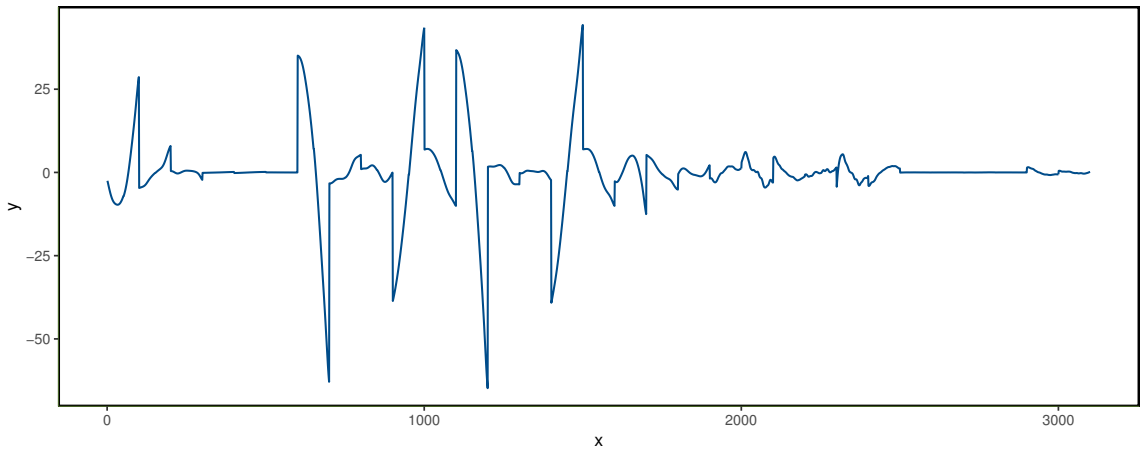


Figure 3.7: Mean centered input sequence for patient 1

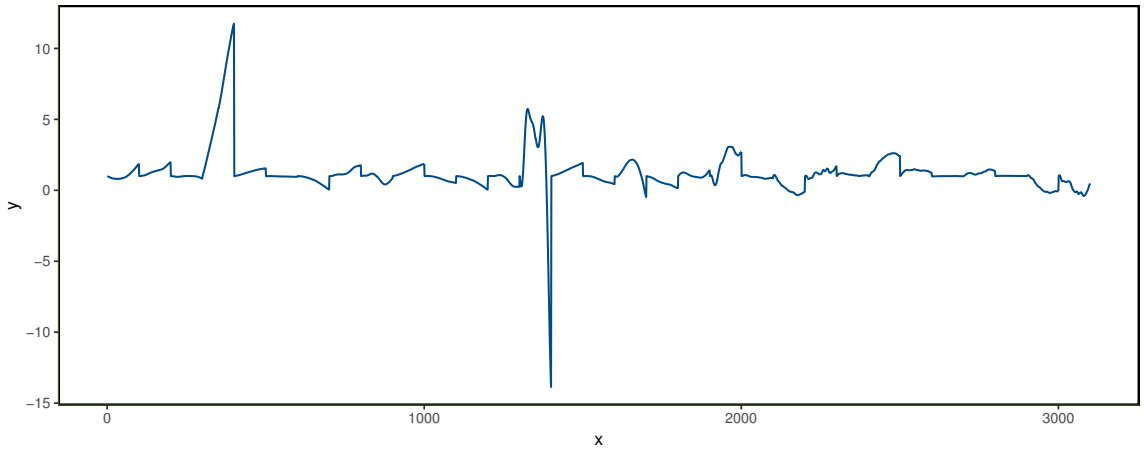


Figure 3.8: Relative variation scaled sequence for patient 1

3.5 Feature Selection

For analyzing the motion quality of THA patients, a meaningful selection of features should be made that corresponds to the most relevant information for a type of motion. For "sit-to-stand" motion, the focus is on the lower extremity of the human body which is the most active part during the motion. Lower extremity governs the forces acting on various parts including joints and muscles.

For human bodies with artificial peripherals embedded in their musculoskeletal system, the focus of interest is on the joint forces and the forces acting on the muscles that function as actuators. The lower extremity joints active during a "sit-to-stand"

motion are hip joints and knee joints, and these joints are connected via the quadriceps femoris muscle group that consists of the following:

- Rectus femoris
- Vastus intermedius
- Vastus lateralis
- Vastus medialis

Accounting the role of the quadriceps femoris muscles as strong extensors in the patellofemoral joint function, Appendix A.2.4, these muscles significantly impact the ability to stand and walk, and thus the motion quality of the human body [18]. The hip joint replacement will primarily affect the post-surgical motion of the body, and the variables specific to the quadriceps femoris muscles will hold higher significance in motion analysis.

3.6 Patient Profiles

Each patient has unique body structure, complimenting that, patients could have different or same symptoms leading to THA and will have different post-surgical consequences on the body movements and forces acting on the body parts. Post-surgical motion can be analyzed by visualizing forces acting on the hip joints, knee joints and the quadriceps femoris muscles collectively called a patient's profile. Based on different patient profiles, different inferences can be observed corresponding to their post-surgical motion quality. Following are the patient-specific discussions for three patients, on how the loads on joints and muscles affect the motion quality by observing the plots of the forces.

Figure 3.9 illustrates forces acting on the joints and muscles of patient 1 during one cycle of "sit-to-stand" motion. It is evident that the magnitudes of forces are ascending just after the patient initiated the activity to stand up from a chair and then gradually descending towards the end of the motion. Overall this is a satisfactory motion as the last state of the body is static standing and we expect the body forces to stabilize as there is no motion in progress.

The downside of this profile is that the maximum magnitude of forces acting on hip joints and knee joints are more than five times the body weight and is more than ten times on the right hip joint. This explains the efforts made by the patient to stand up

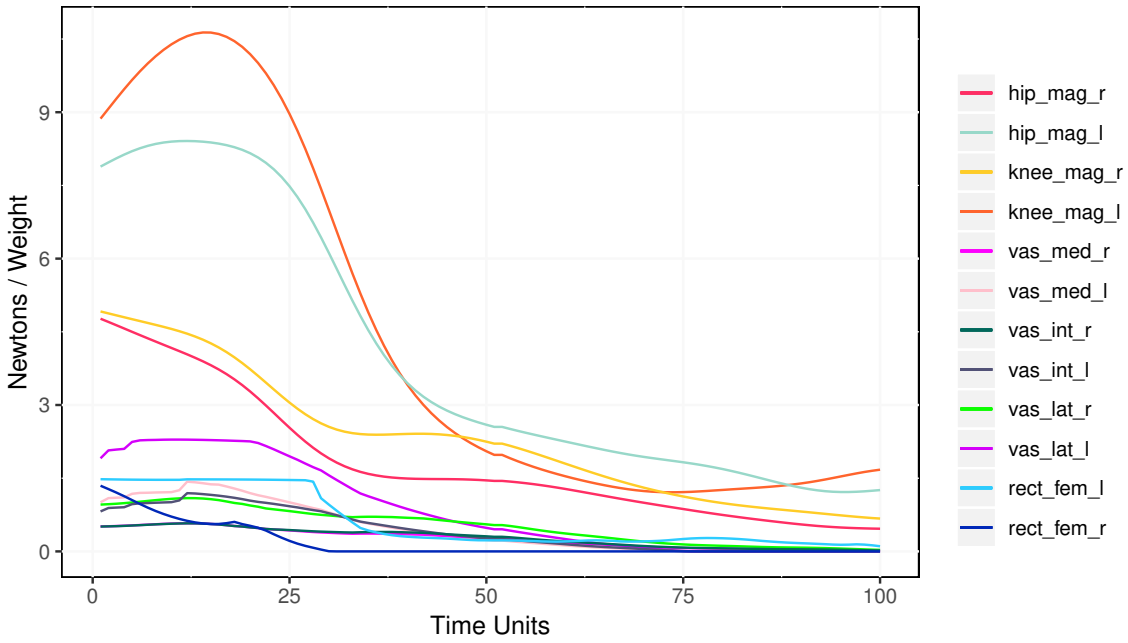


Figure 3.9: Patient 1 profile with selected features

from a static sitting position. From this profile, an imbalance in the lower extremity is noticeable by looking at the peaks of hip joint forces acting on both sides. There is a considerable difference in the forces acting on both the hip joints and is a cause of imbalance which affects the overall motion quality.

Figure 3.10 illustrates forces acting on the joints and muscles of patient 32 during one cycle of "sit-to-stand" motion. This profile is different from that of patient 1 as, during the start of the motion, the magnitudes of all forces are less than 3 times the body weight, that suggests less efforts made to stand up from a chair, but after 40 time units, the forces have increased significantly for all the features and are inconsistent. When the patient is in a static standing position, the forces are not normalized and are at higher peaks as compared to when the motion started. Patient 32 is more comfortable in a sitting position and requires additional efforts to stand. This is possibly a case of an unsatisfactory hip replacement surgery because, during static standing position, the body is still applying forces to attain an equilibrium. Moreover, the imbalance is evident from this profile as the magnitudes of forces on all features on both sides respectively are significantly different.

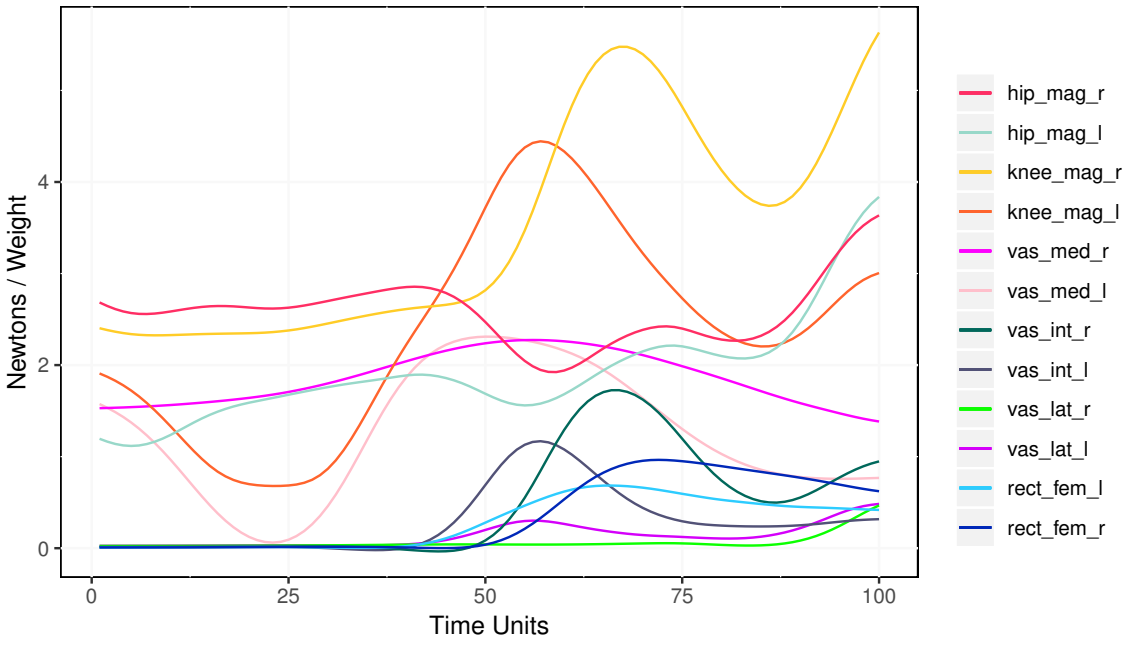


Figure 3.10: Patient 32 profile with selected features

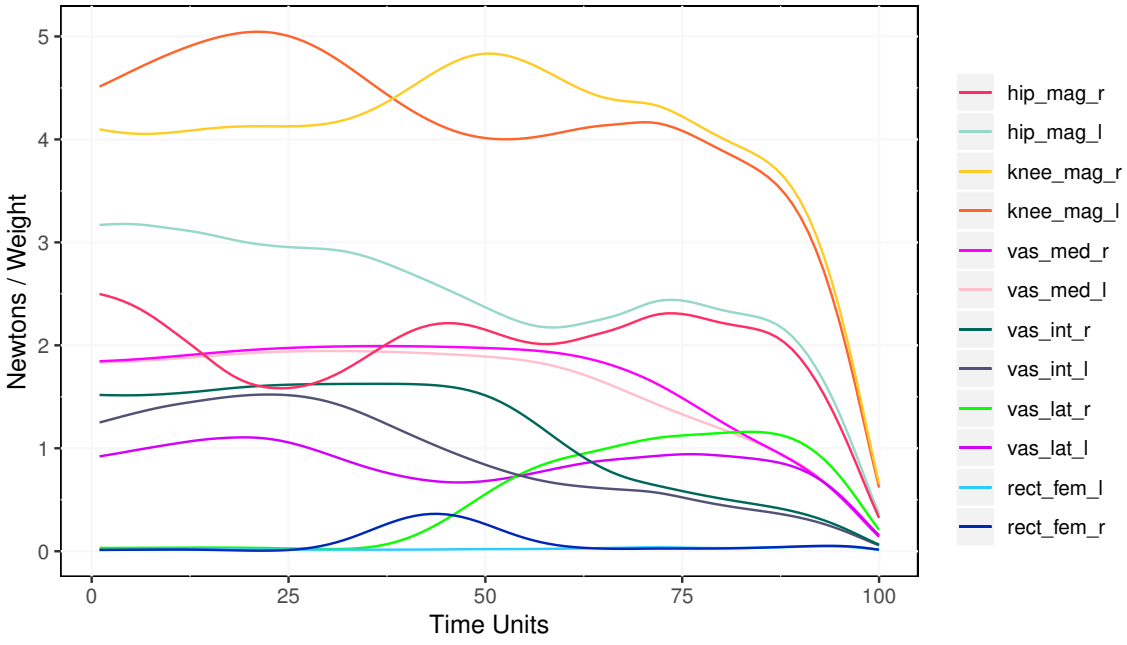


Figure 3.11: Patient 46 profile with selected features

Figure 3.11 illustrates forces acting on the joints and muscles of patient 46 during one cycle of "sit-to-stand" motion. This profile suggests a difficult start of the motion

while standing up from a chair as the magnitudes of forces acting on joints and some muscles are higher during the start of the motion. Similar to the other patients, the forces acting on the hip and knee joints are considerably higher than the load on the muscles.

This profile has a converging pattern of forces where the motion start was not as good as the end of the motion. The patient's standing stance is quite stable as all the forces are converging to their minimum magnitudes at the time step 100. This patient has a satisfactory static position, but when the body is in motion, the forces are imbalanced and seem to be a laborious task.

3.7 Feature Contrast

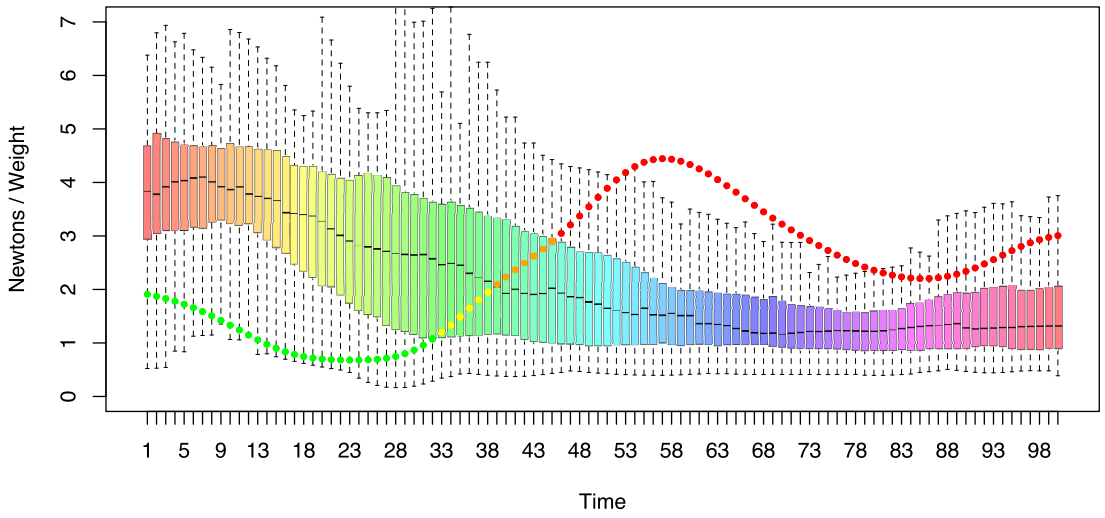


Figure 3.12: Contrasting forces on left knee joint of patient 32 with other patients

To establish a relation between a specific patient and all other patients to analyze the relative motion quality, we constructed an analysis shown in Figure 3.12 that represents box plots of a feature of 47 patients at each time step, contrasting the same feature of a specific patient with a time series illustrated using dots. It can be seen that where, during the motion, the force acting on a feature is higher or lower as compared to the other patients' forces.

Figure 3.12 is a contrast between patient 32 and all other patients, illustrating the distribution of force acting on the left knee joint. The force on the joint is shown using dots which lies in every zone of the overall distribution of the other 47 patients.

The points in the zone between the first(0%) and second quartile(25%) are colored with green and are lying in the safe zone as these loads are lower than the average loads of other patients.

The points lying between the second(25%) and fourth quartile(75%) are marked with yellow (for points less than the median(50%)) and orange (for points greater than the median(50%)). This zone is the average zone since, in this zone, the load on the left knee joint is in the average magnitude interval of load distribution of all other patients.

Points that are between fourth quartile(75%) and fifth quartile(100%) are colored in red, as they lie in the zone where points exceed the average magnitudes of the force acting on the same feature for other 47 patients at each time step. Any point lying in this zone contributes to a poor motion quality.

This approach of feature contrast is used for analyzing all the features of the 48 THA patients to understand their "sit-to-stand" motion profiles with more insights about load patterns and relative motion qualities.

Chapter 4

Research and Analysis - Phase 1

4.1 Introduction

In this section, we attempt to analyze the patients' profiles by visualizing the forces acting on the joints and muscles of the lower extremity during one cycle of the "sit-to-stand" motion and discuss how it can affect a patient's motion quality. We also explain how we can create a motion quality metric using the available dataset of muscle and joint forces for the motion. Further, we discover that the motion quality, quantified by remodeling the muscle and joint forces data, is embedded in the inverse kinematics dataset by visualizing the cluster formation with the help of component analysis techniques like PCA and PARAFAC. The discovery of motion quality in the inverse kinematics dataset leads to the motivation of improving the motion quality assessment and recreating the metric of patients' distribution in the motion quality categories: "Fair" and "Poor". This section also explains about some data remodeling techniques that can be utilized while working on a wide-shaped dataset.

Figure 4.1 shows the flow of research and analysis we perform to achieve a set of relevant features from the inverse kinematics dataset and muscle forces that resemble the body balance during the "sit-to-stand" motion. These features are further used with a regression model and ANN to predict the classified motion quality, as explained in Chapter 5.

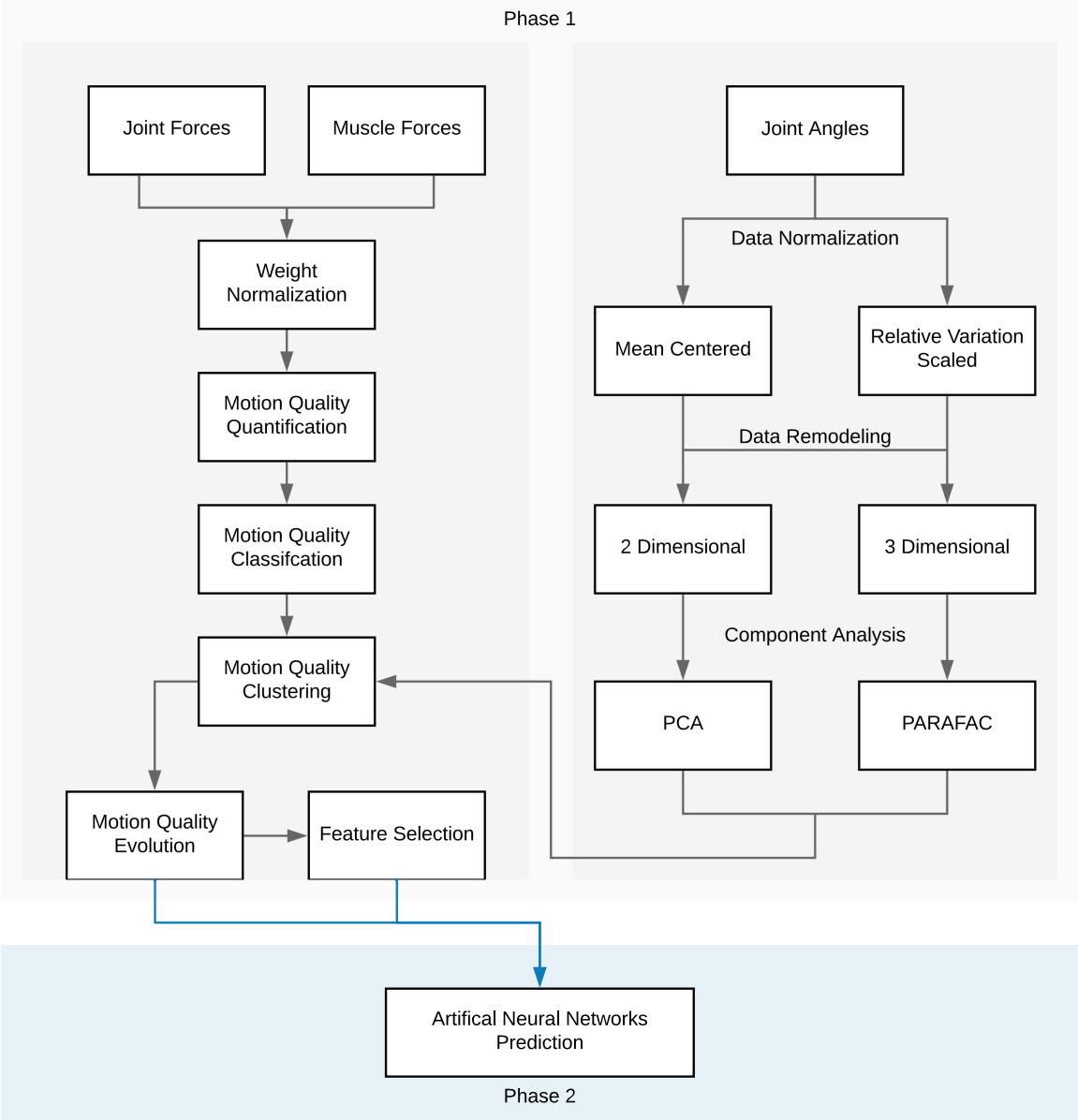


Figure 4.1: Research and Analysis flow chart, Phase 1

4.2 Motion Quality Analysis

The profile of patients is used to categorize the motion quality by analyzing the loads on the muscles and joints of both sides of the lower extremity. The profiles explain the inconsistency in the load patterns of muscles and joints, which is a sign of instability of motion. Also, there are different ranges of high load magnitudes at

different time steps which tells about efforts exerted by the body during various stages of a "sit-to-stand" motion.

To address the instability of forces during the motion and inconsistent load patterns together, we use the difference in the magnitude of each feature of both sides of the human body for motion quality analysis. The differences, which are referred to as deltas, allow the consideration of discrepancies in muscle and joint loads on both sides of the human body. In this way, the body balance can be analyzed by looking at load dominance on either side of the lower extremity during one cycle of the motion.

Algorithm 4 Feature Delta Calculation

Input: $F_{right},\ F_{left}\ for\ P_1 \dots P_{48}$
Output: *Feature Delta Series for all Patients*

```
1: function CalculateFeatureDelta(F_right, F_left)
2:     Let δ be a 2D vector
3:     for p ← 1 to 48 do
4:         for t ← 1 to 100 do
5:             δ[p, t] = F_right[p, t] − F_left[p, t]
6:         end for
7:     end for
8:     return δ
9: end function
```

Algorithm 4 is used to calculate the delta of a feature for all the patients by subtracting the loads on the left side from the loads on the right side of the lower extremity.

Figure 4.2 represents a delta data of the vastus medialis muscle load for patient 29, which is contrasted with the other 47 patients' delta distribution at each time step. It has two parts :

1. Collective box plot
2. Muscle load delta dot plot

Collective box plot

The box plot shown in Figure 4.2 represents the distribution of vastus medialis muscle load for the other 47 patients at each time unit. There are a total of 100 box plots for a complete motion cycle, and for each time unit, they represent a distribution

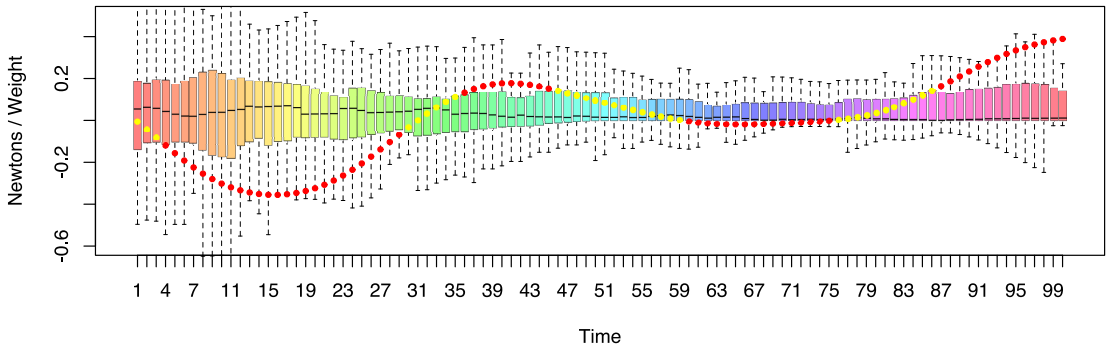


Figure 4.2: Vastus Medialis Load Delta, Patient 29

of delta force acting on the muscle for every patient except patient 29. This plot is used to compare the load dominance of patient 29 by contrasting with the delta series of the same delta feature of all the other 47 patients.

Muscle load delta dot plot

The curve represented by the dots is the delta series of vastus medialis muscle load of patient 29 that describes a pattern of load dominance on the muscles of the left and right side of the patient. It is evident that during the "sit-to-stand" motion, the load dominance oscillates from one side to the other. As these delta values are a result of subtraction between the loads on two muscles, it represents a continuous change in the load dominance moving from one side to another. The plot shows that during the motion, at different stages, the patient shifts the load to either side to attain a balance to get up from the chair and stand.

Unlike the feature contrast in Figure 3.12, there are two "dominant" regions in Figure 4.2, one less than the second quartile and one greater than the fourth quartile. Any point lying in either of the regions represents the load dominance on either side of the lower extremity.

By comparing the delta plot of patient 29 with all other patients' load distribution, we infer about the dominance of load on each side. At each time step, if the delta force has a magnitude less than the second quartile value, the force is dominant on

the muscle of the left side, and if it is greater than the fourth quartile, the force is dominant on the muscle of the right side.

The "dominant" region is in the outer region of the average interval of the load distribution of all other patients at each time step, and we are interested in the occurrences of the load in that region. Any point lying in the "dominant" region accounts for the degraded quality of motion as it represents load imbalance in the lower extremity during the motion cycle.

4.3 Motion Quality Quantification

To quantify the motion quality of the patients for a "sit-to-stand" motion, selection of features that are most relevant is done from the features of inverse dynamics and muscle forces. As discussed in the section for feature selection, Section 3.5, the quadriceps femoris muscles are the significant entities of the musculoskeletal system to focus on to perform the motion analysis. Hip joints and knee joints are also substantial and active parts of the body while performing the "sit-to-stand" motion. Hence, the following features are selected among all other time-series features, for the initial analysis and quantification of the motion quality:

1. Rectus femoris force
2. Vastus medialis force
3. Vastus intermedius force
4. Vastus lateralis force
5. Knee loint force
6. Hip joint force

Let the above set of features be the Feature Set 1. This selection of variables covers the forces acting on the hip and knee joints and the muscles connecting them.

Finally, the quantification of the relative motion quality for each patient is performed using Algorithm 5 by counting the delta force occurrences in the "dominant" region for all the delta series that constitutes Feature Set 1. Figure 4.3 represents the result of the motion quality quantification for each patient.

Figure 4.3 represents the percentage of total "dominant" region occurrences for each patient for 6 features in Feature Set 1 that can have total 600 "dominant"

Algorithm 5 Motion Quality Quantification

Input: δ_{F1} ... δ_{Fn} *for* $P_1 \dots P_{48}$
Output: *Quantified Motion Quality Array* σ

```
1: function QuantifyMotionQuality(δ_F[])
2:     Let σ be a 1D vector
3:     for p ← 1 to 48 do
4:         for δ ← δ_F[] do
5:             δ_p ← δ[p,]
6:             δ_quartile ← GetQuartiles(δ[−(p),])
7:             for t ← 1 to 100 do
8:                 if (δ_p[t] < δ_qaurtile[2,t]) Or (δ_p[t] > δ_qaurtile[4,t])
9:                 then σ[p] + = 1
10:            end for
11:        end for
12:    end for
13:    return σ
14: end function
```

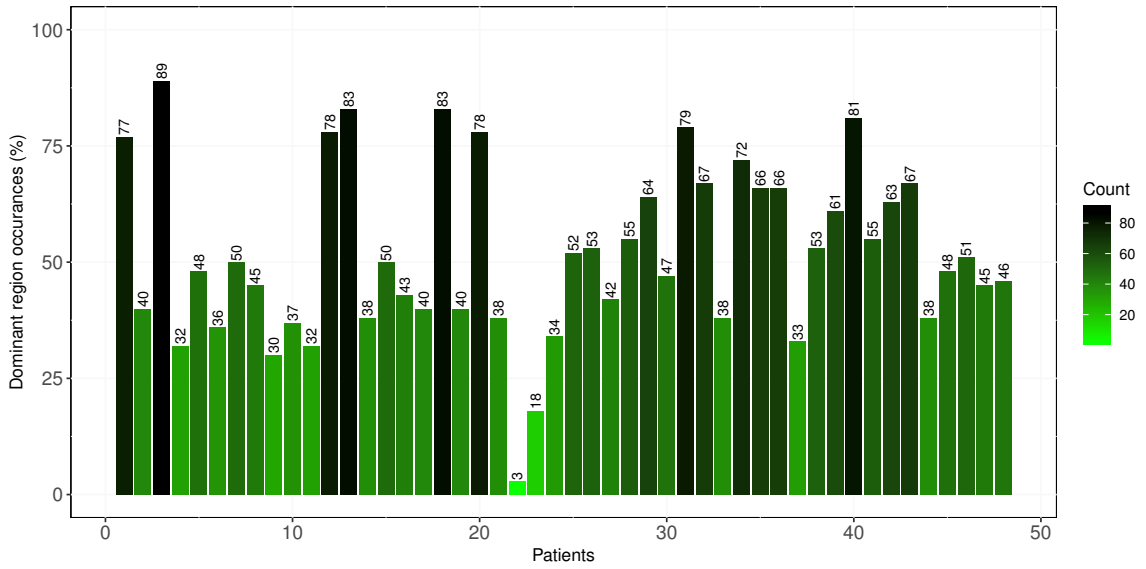


Figure 4.3: Dominant section occurrences for all the patients using the Feature Set 1

region occurrences. It can be seen that the highest percentage of "dominant" region occurrences, which is 89%, is for patient 3. To understand it more, Figures 4.4 and 4.5 illustrate the profile of patient 3 by contrasting the delta features in Feature Set 1 with other patients. Most of the "dominant" region occurrences have a negative

magnitude which means that the load dominance is on the left of the lower extremity throughout the "sit-to-stand" motion. It may be inferred from this profile that the THA would have been done on the right hip joint as the patient prefers to put more load on the left side. The consequence of deliberate dominance on either of the sides affects the motion quality significantly.

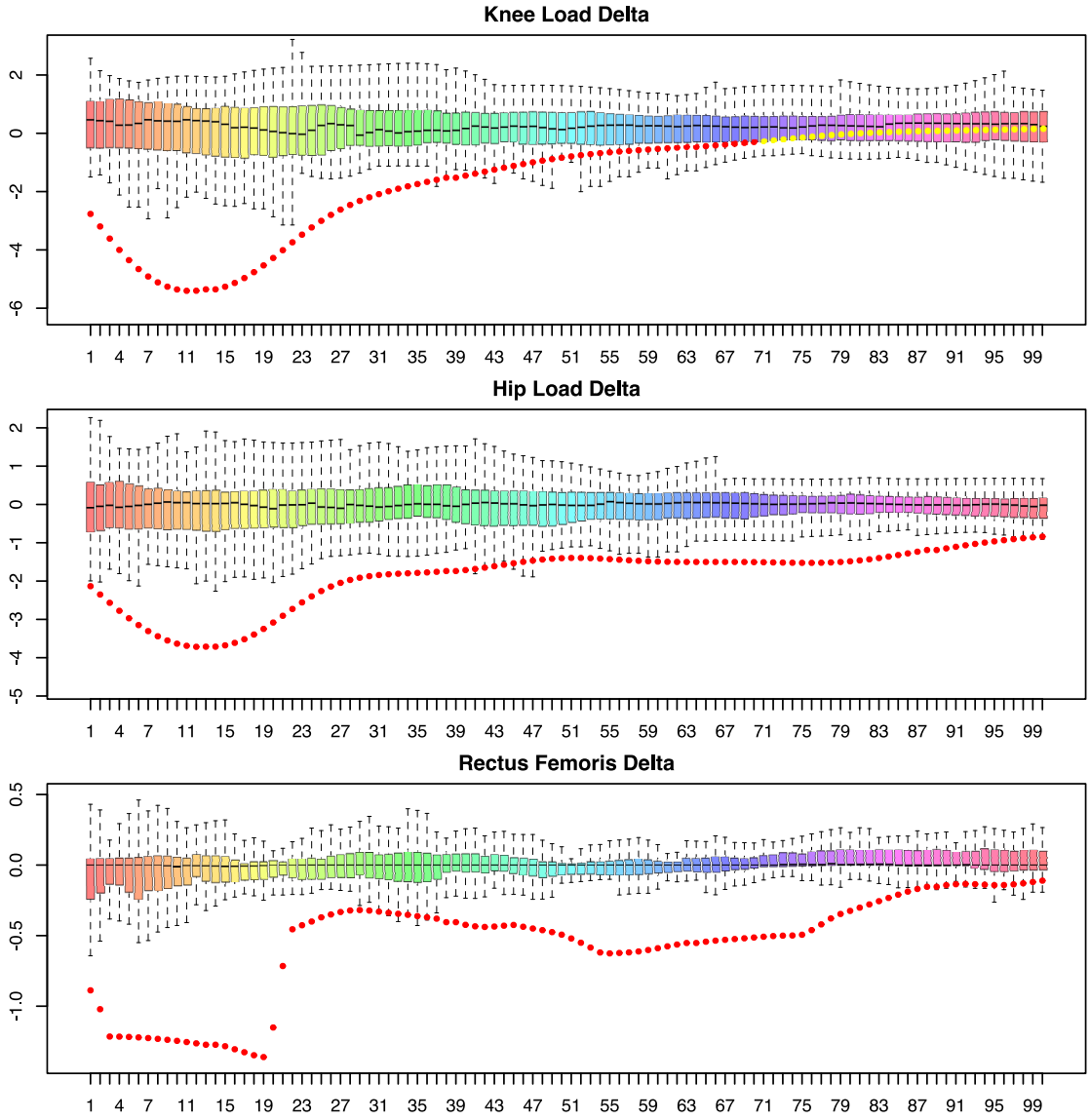


Figure 4.4: (a) Feature delta contrast for patient 3 using Feature Set 1

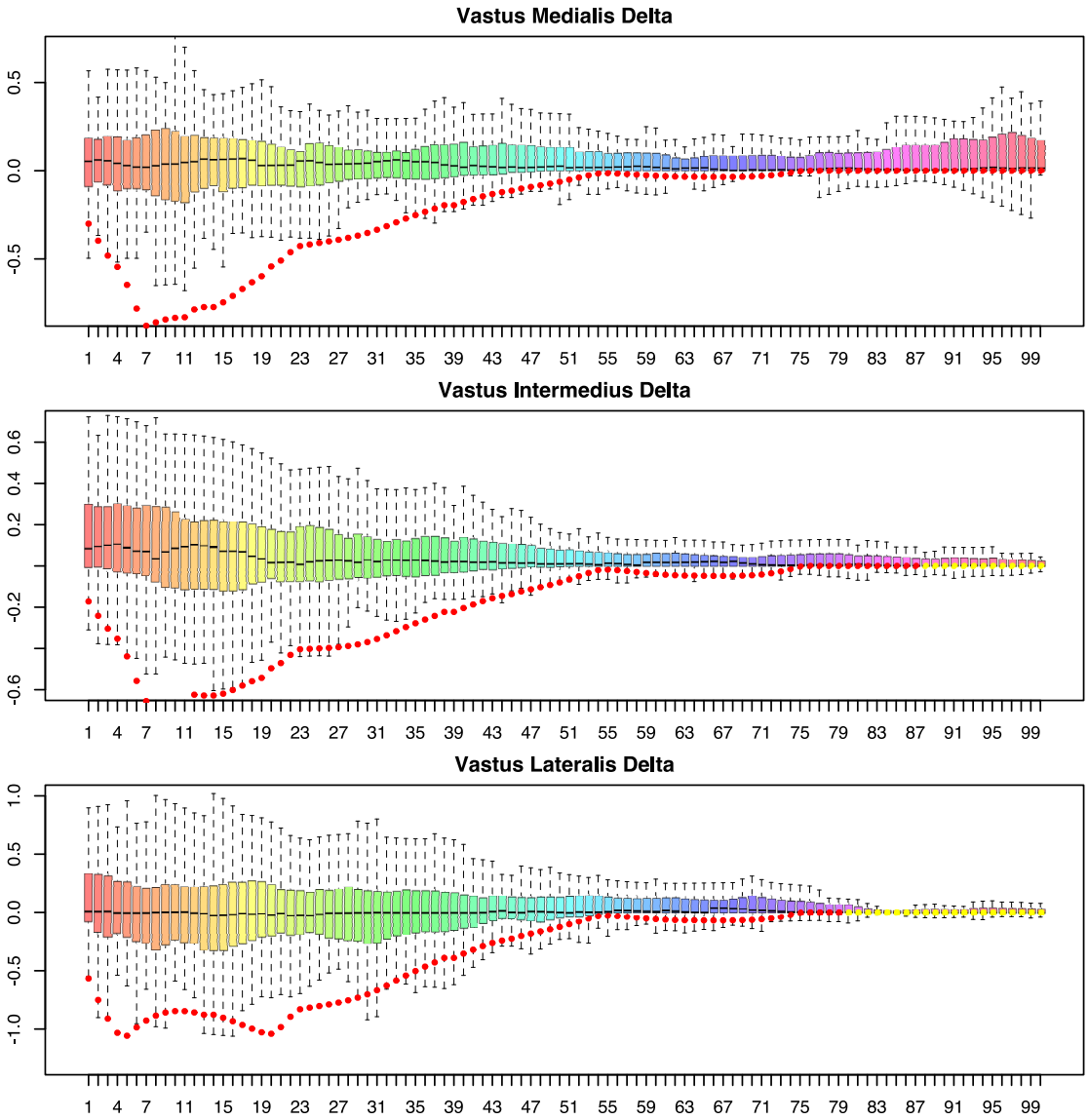


Figure 4.5: (b) Feature delta contrast for patient 3 using Feature Set 1

4.4 Motion Quality Classification

The quantified motion quality, in Figure 4.3, is a continuous variable where every patient can have the "dominant" region occurrences ranging from zero to 600 (6 features with 100 time-series each). The continuous variable is converted to a categorical variable to classify the motion quality.

The motion quality of THA patients is affected by the surgery as the prosthetic implants don't perform as natural hip joints; therefore, the motion quality is classified

into the categories: "Fair" and "Poor". The "Fair" motion quality category suggests that there is an irregular pattern of forces acting on the muscles that cause some immobility in the motion, whereas "Poor" motion quality indicates that possibly the THA surgery was not satisfactory for the patient and had affected the movements with a high degree of instability.

We chose only two classes for motion quality because increasing the number of classes will decrease the number of patients falling in each category. The data we use is for 48 patients, which is a small dataset for machine learning algorithms.

Algorithm 6 is used to classify the patients in the categories of motion quality, which divides the patients based on the count of occurrences in the "dominant" region into a given number of categories. The numerical values calculated for the categories "Fair" and "Poor" are 0 and 1 respectively.

Algorithm 6 Motion Quality Classification

Input: $Quantified\ motion\ quality\ vector\ \sigma$
Output: $Motion\ quality\ classification\ vector$

1: **function** MOTIONQUALITYCLASSIFIER($\sigma[], Categories, FeatureCount$)
2: $Let\ QualityVector = []$
3: **for** $count \leftarrow \sigma$ **do**
4: $$QualityVector[i] = \left\lfloor \frac{count}{(FeatureCount * 100)} * Categories \right\rfloor$$
5: **end for**
6: **return** $QualityVector$
7: **end function**

Table 4.1 shows the distribution of patients into the categories "Fair" and "Poor". From this analysis, we can divide the patients into two groups, where the patients in group "Fair" are more than the patients in group "Poor". This classification is further used to analyze the inverse kinematics dataset to observe any information that can be useful to form clusters that separate the patients based on these categories.

Fair	Poor
26	22

Table 4.1: Distribution of patients in motion quality categories using Feature Set 1

4.5 Principal Component Analysis (PCA)

For the initial analysis, we have used all the features of the inverse kinematics dataset and tried to visualize the components generated by the PCA algorithm with classified motion quality to see any cluster formation of both classes respectively.

This technique requires the input dataset in the form of a 2-dimensional vector. Hence, mean centered and relative variation scaled data is used for PCA separately to determine the suitable dataset among the two for the motion quality classification. All the features are stacked one after the other for all patients in separate rows that form a 2-dimensional vector.

There is a total of 19 time-series features with 100 time units each in the inverse kinematics dataset for 48 THA patients' observations. Therefore, according to the rule of maximum components, discussed in Section 2.5.1, the maximum number of components that can be generated by the PCA is 48.

4.5.1 PCA - Inverse Kinematics

Mean Centered

The mean centered time-series, as shown in Figure 3.7, is used for the analysis. Figure 4.6 shows variance in each component and a cumulative variance on the top of each bar plot. It is evident that the variance in each component is decreasing exponentially from the first to the forty-eighth component. The cumulative variance for the last component is 100%, which shows that all the components cumulatively represents the variance in the original inverse kinematics dataset.

Each component has 48 data points that represent the original data points for each patient. Figure 4.7 is the representation of PC1 and PC2 where the dots and triangles represent patients classified in the motion quality categories "Fair" and "Poor" respectively. From this plot, it is observable that the patients in both categories are grouped in their respective clusters. There is a high amount of intersection between both the clusters that causes misclassifications, and hence, the analysis using motion quality from the Feature Set 1 and all inverse kinematics features can be improved.

Result: The cluster formation of the two classes is a good sign which represents the existence of motion quality information in the inverse kinematics dataset. In Figure 4.7, PC1 is the component which separates the two clusters and according

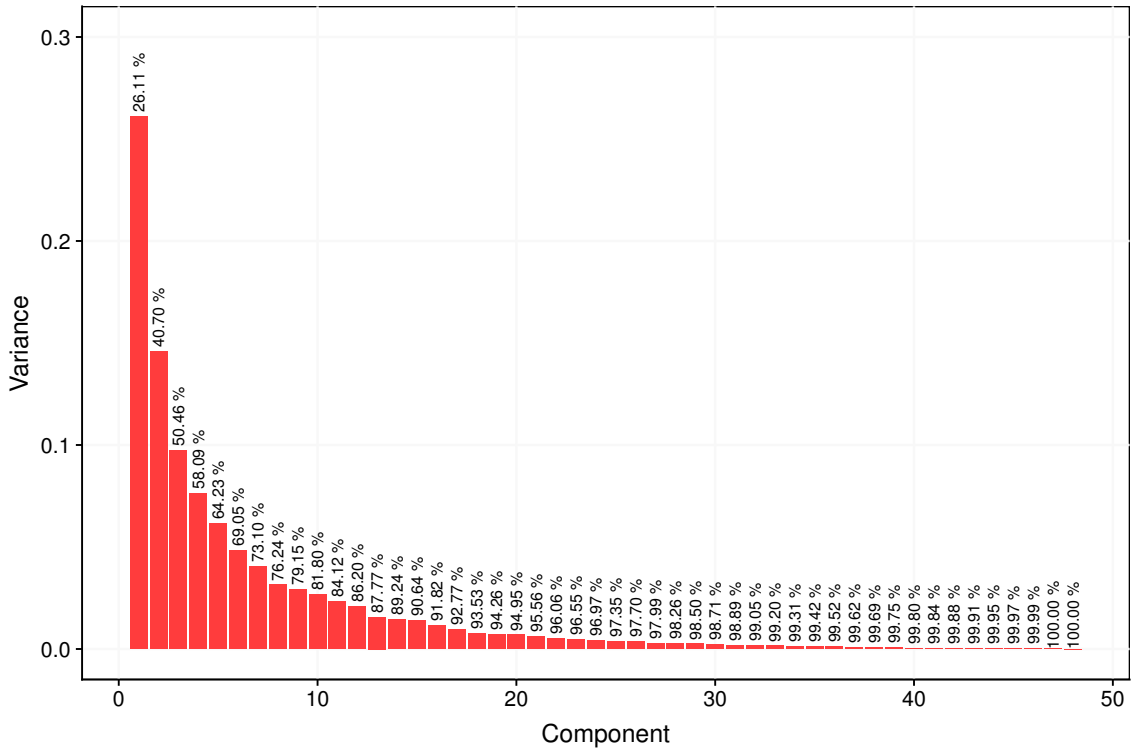


Figure 4.6: Variance in the components generated by the PCA on the mean centered inverse kinematic features

to Figure 4.6, PC1 accounts to 26.11% of the variance in the original dataset. The projected data points of each patient on the PC1 are in such an order that forms a grouping according to the motion quality categories. Hence, PC1 becomes the most important component out of the 48 components that also accounts to the maximum variance of the original dataset than others. From this result, we can infer that the motion quality does exist in the inverse kinematics dataset with a significant amount of variance.

In the next steps, we try to improve the motion quality quantification and select the relevant inverse kinematics features for component analysis that can improve the clustering significantly.

Relative Variation Scaled

The relative variation scaled time-series, as shown in Figure 2.7, is used for the analysis. Figure 4.8 shows variance in each component and a cumulative variance

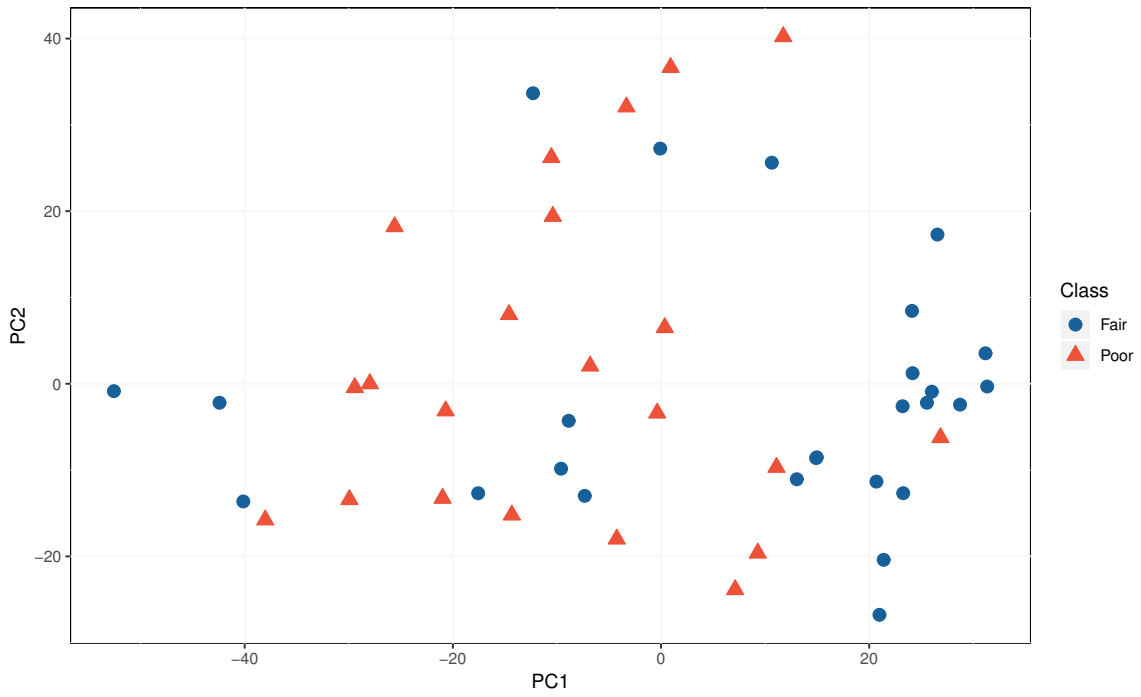


Figure 4.7: Clustering of the patients using motion quality categories using the Feature Set 1, and components generated using all the mean centered inverse kinematics features

on the top of each bar. It is observable that the variance is decreasing exponentially from the first component to the forty-eighth component. The cumulative variance for the last component is 100%, which shows that all the components cumulatively represents the variance in the original inverse kinematics dataset.

Figure 4.9 is the representation of PC1 and PC2 where the dots and triangles represent patients classified in the motion quality categories "Fair" and "Poor" respectively. This plot depicts that data points in both the classes are grouped in their respective clusters. There is a high amount of intersection between both the clusters, and hence, the analysis using motion quality from the Feature Set 1 and all the relative variation scaled inverse kinematics features can be improved.

Result: The cluster formation using relative variation scaled inverse kinematics data has a high degree of the intersection. In Figure 4.7, PC1 is the component which separates the two clusters and according to Figure 4.6, PC1 accounts to 22.84% of the

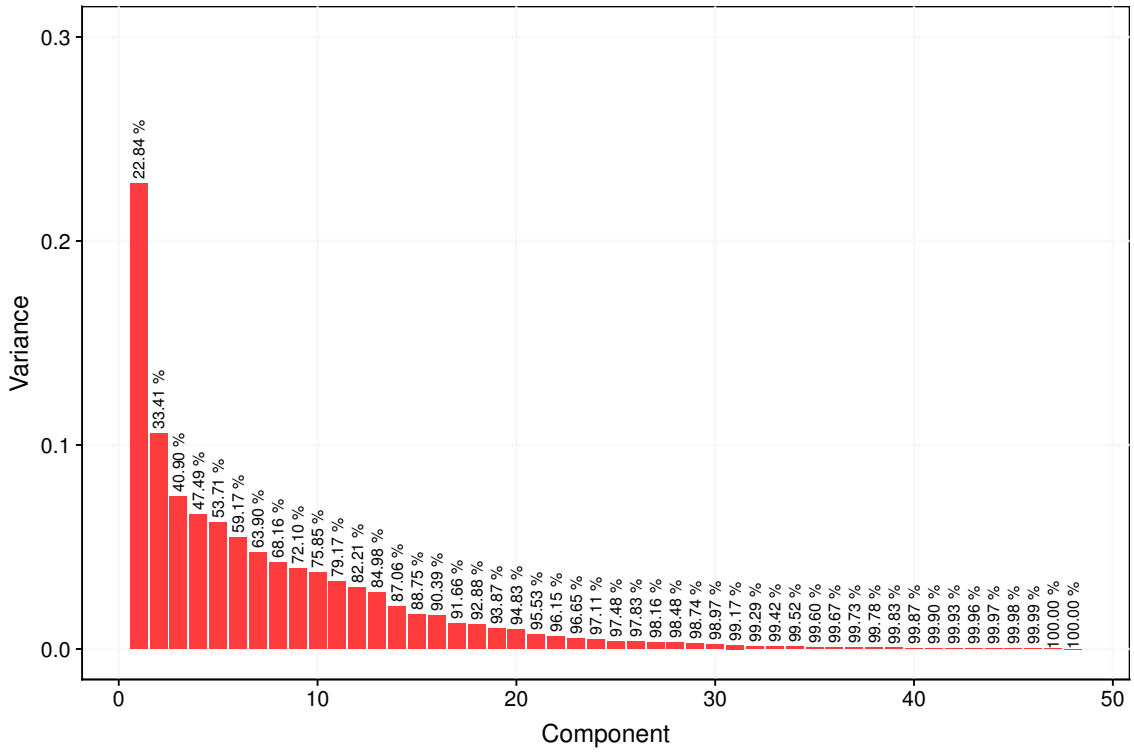


Figure 4.8: Variance in the components generated by the PCA on the relative variation scaled inverse kinematic features

variance in the original dataset. From this result, we infer that the relative variation scaled dataset can be used for further improvements in the motion quality analysis.

PCA Result: Motion Quality Discovery

The inverse kinematics dataset can classify the patients in their respective motion quality categories, which were derived by the delta features of joint loads and muscle loads. Both the mean centered and the relative variation scaled dataset produced clustering where the mean centered has better cluster separation.

The result so far is not satisfactory as there are many misclassified patients in the clustering. Possibly, the Feature Set 1 has unnecessary information about the musculoskeletal forces that do not resemble the balance of the human body during the "sit-to-stand" motion.

For the process of improving the motion quality quantification, in the next steps, we form different feature sets that represent information about different parts of the

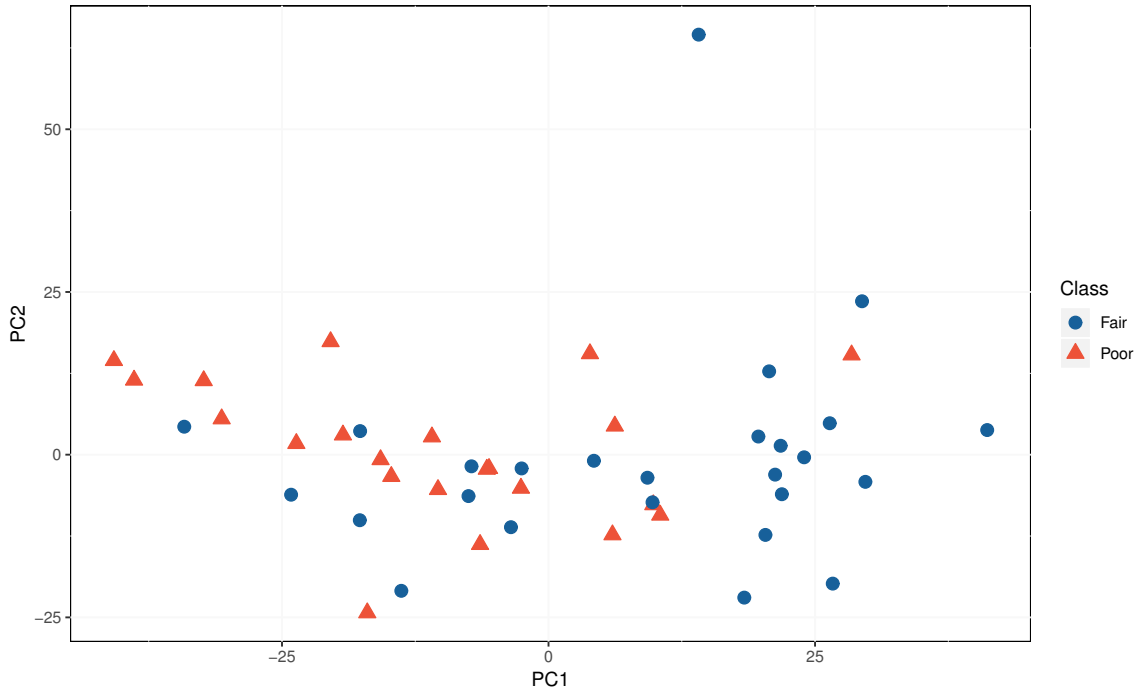


Figure 4.9: Clustering of the patients using motion quality categories using the Feature Set 1, and components generated using all the relative variation scaled inverse kinematics features

musculoskeletal system. We aim to find the set of features from the muscle and joint forces that can accurately resemble the body balance, which in turn will improve the cluster formation.

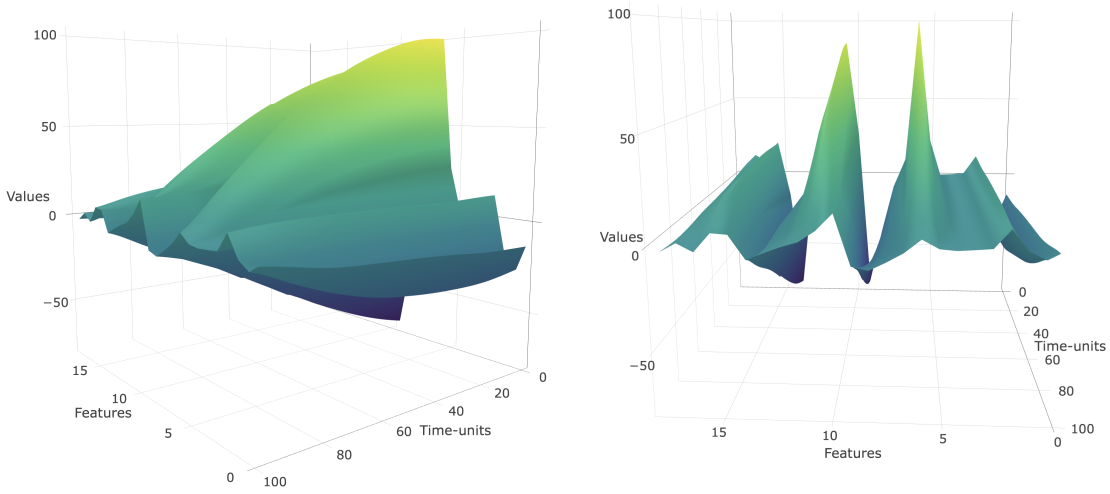
4.6 Parallel Factor Analysis (PARAFAC)

PARAFAC is another technique for dimensionality reduction that we use to explore and analyze the motion quality of all the patients in both the normalized variants of the inverse kinematics dataset. PARAFAC works on a multi-way dataset, and therefore, a 2-dimensional vector of inverse kinematics is remodeled into a 3-dimensional vector.

4.6.1 Data Remodeling

The inverse kinematics dataset is logically a 3-dimensional dataset where the dimensions are patient's index, features, and time-series values of the features. Figure

4.10 represents the 3-dimensional surface for the remodeled original inverse kinematics dataset for patient 1.



(a) 3D viewpoint 1

(b) 3D viewpoint 2

Figure 4.10: 3D model of original inverse kinematic features for patient 1

Mean Centered

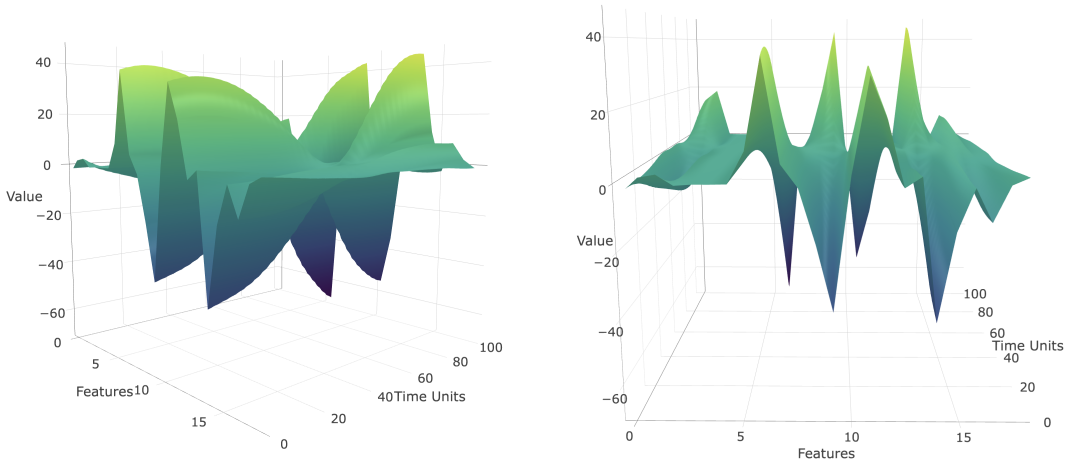
Figure 4.11 is the 3-dimensional representation of mean centered inverse kinematics dataset for patient 1, which is used for the PARAFAC analysis.

Figure 4.12 represents clustering formed by the two components extracted from PARAFAC on a mean centered inverse kinematics dataset with motion quality quantified using Feature Set 1. There are two clusters in the plot with a high degree of mixed data points of each category. This analysis with 3-dimensional data does not yield the information that we seek because the remodeled dataset is prone to loss of information pertaining to the classification of motion quality.

Relative Variation Scaled

Figure 4.13 is the 3-dimensional representation of relative variation scaled inverse kinematics dataset for patient 1 which is used for the PARAFAC analysis.

Figure 4.14 represents clustering formed by two components extracted from the PARAFAC analysis on relative variation scaled inverse kinematics dataset with motion quality quantified using Feature Set 1. The clustering for both the classes has



(a) 3D viewpoint 1

(b) 3D viewpoint 2

Figure 4.11: 3D visualization of mean centered inverse kinematic features for patient 1

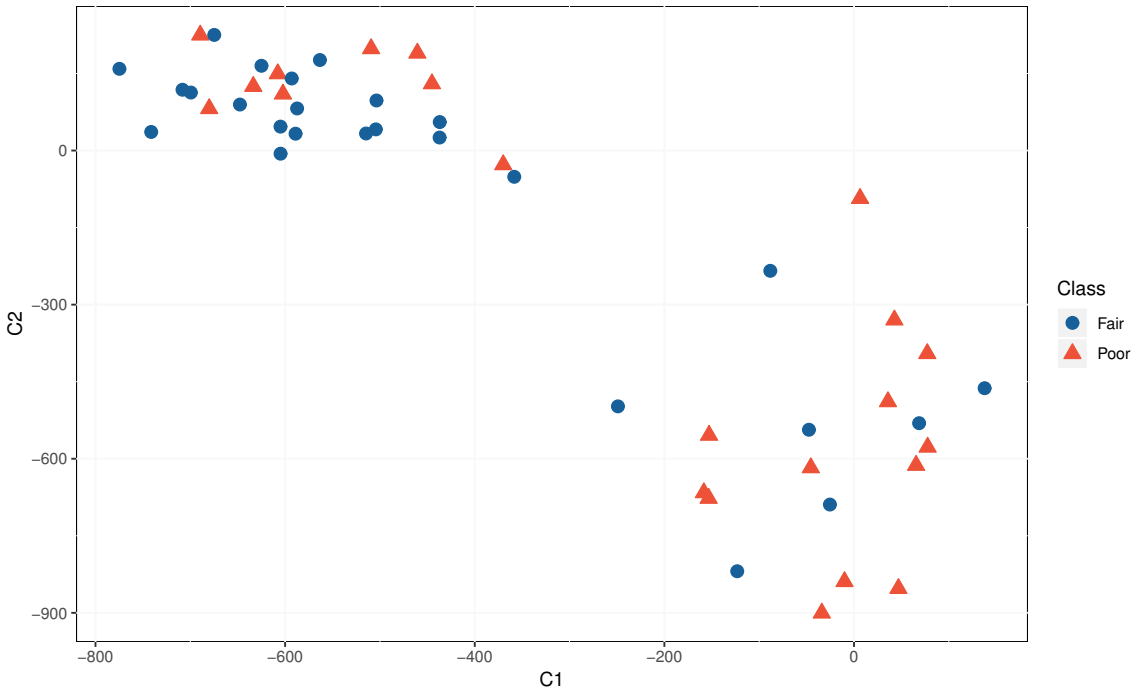
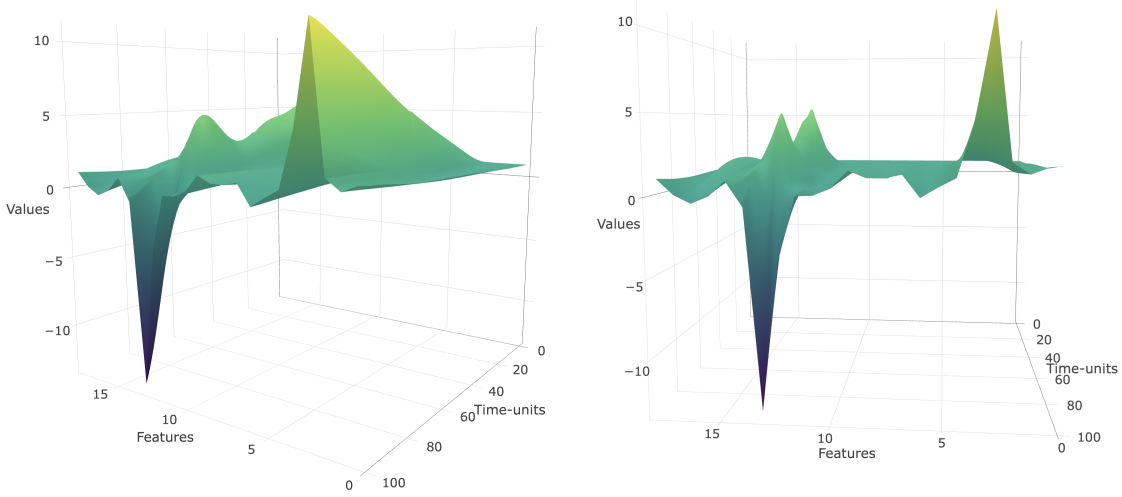


Figure 4.12: Clustering of the patients using motion quality categories using the Feature Set 1, and PARAFAC components generated using all the mean centered inverse kinematics features



(a) 3D viewpoint 1

(b) 3D viewpoint 2

Figure 4.13: 3D model of relative variation scaled inverse kinematic features for patient 1

a high degree of overlap. Therefore, PARAFAC analysis on relative variation scaled dataset is not a suitable option for further analysis.

4.7 Motion Quality Evolution

With the help of PCA and PARAFAC, we are to able to achieve the dimensionality reduction of the dataset from 600 attributes to 48 attributes called principal components. Each patient has it's unique projected location on every component. Therefore, for a type of dataset, for example, mean centered dataset, the relative position of each data point in Figure 4.7 is fixed. To achieve better clustering of the patients in their respective motion quality categories, represented by the labeling of the data points in Figure 4.7, we aim to improve the motion quantification process by refining the selection of features used to calculate the motion quality metric.

From the previous analysis, we discovered that PCA with mean centered dataset has produced a good clustering of motion quality quantified using Feature Set 1 as compared to the analysis on relative variation scaled inverse kinematics dataset. On the other hand, PARAFAC has also produced better clustering with mean centered dataset as compared to the relative variation scaled dataset. Therefore, we perform all

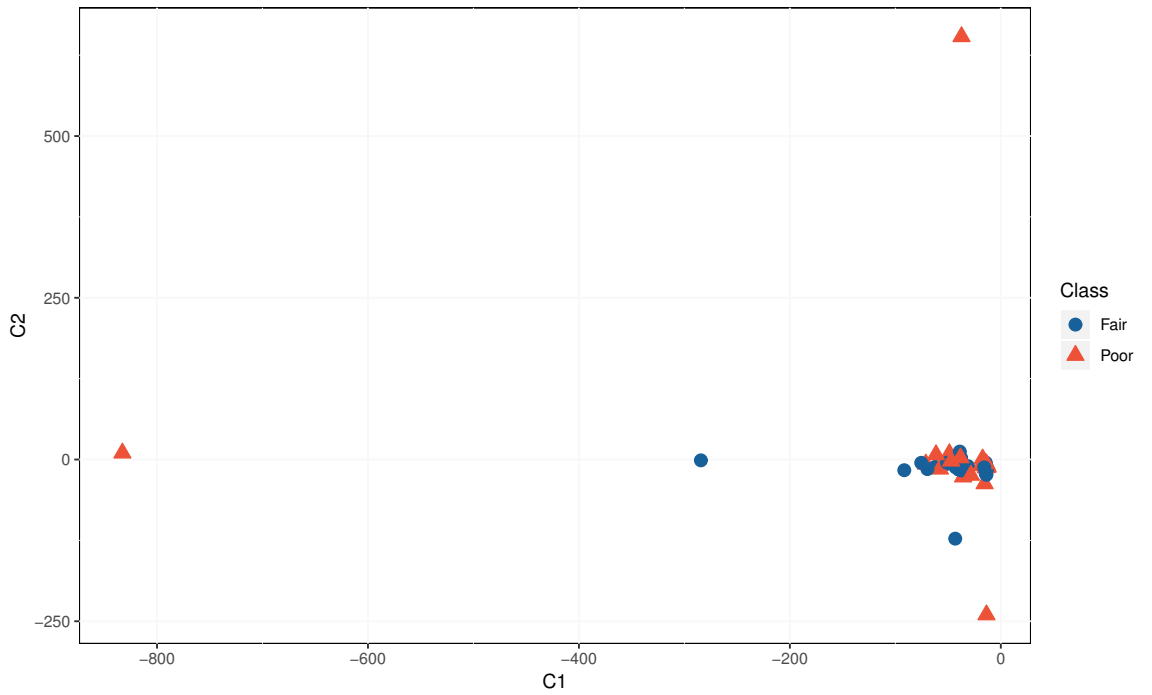


Figure 4.14: Clustering of the patients using motion quality categories using the Feature Set 1, and PARAFAC components generated using all the relative variation scaled inverse kinematics features

possible analysis using PCA and PARAFAC on mean centered, and relative variation scaled dataset with new feature sets, to ensure the achievement of the best clustering for motion quality.

The previous analysis has proved that the body balance can be detected using two classes "Fair" and "Poor", as the component analysis techniques produced separate clusters of each class, but with the intersection between the clusters, which can lead to misclassifications. Previous analysis inspires the idea of quantifying motion quality using different feature sets. It infers that the motion quality quantified using Feature Set 1 may not be the best feature set that resembles body balance during the "sit-to-stand" motion. Therefore, we develop different kinds of feature sets that may be related to body balance during the motion cycle. The confirmation of the most relevant feature set is achieved by performing component analysis techniques and visualizing the motion quality classification using respective feature sets. Different feature sets developed are as follows:

1. Feature set 2 : All muscle loads
   (a) Rectus femoris delta force
   (b) Vastus medialis delta force
   (c) Vastus intermedius delta force
   (d) Vastus lateralis delta force

   This feature set contains only the delta forces on muscles included in the quadriceps femoris muscle group. As the muscles acts as actuators to control the motion, this set of muscles can be a good set to analyze the motion balance.
2. Feature Set 3 : Selected muscle loads
   (a) Vastus medialis delta force
   (b) Vastus intermedius delta force
   (c) Vastus lateralis delta force

   This feature set is similar to the Feature Set 2 but excludes the rectus femoris muscle which may not be the most important muscle to be considered while analyzing the body balance.
3. Feature Set 4 : Joint loads
   (a) Hip joint delta force
   (b) Knee joint delta force

   This feature set contains only the forces on the joints of the most active part of the lower extremity during the "sit-to-stand" motion. Considering only the joint loads could produce interesting results.
4. Feature Set 5: Vastus medialis delta force
5. Feature Set 6: Vastus intermedius delta force
6. Feature Set 7: Vastus lateralis delta force

   The Feature Sets 5, 6, and 7 have only one muscle that will help us understand if they are the most important features pertaining to the body balance.
7. Feature Set 8: Hip joint delta force
8. Feature Set 9: Knee joint delta force

   The Feature Sets 8 and 9 have only one joint load that will help us understand the significance of the hip and knee join loads for the analysis of motion quality.

Performing analysis on the above feature sets helps identify any noise we have included in the Feature Set 1, as the new feature sets contain selective features and exclude different sets of features which could be the source of the noise.

Figures in Appendix A.4 represents clustering formed by PCA on mean-centered inverse kinematics dataset and motion quality classified using all the new feature sets. Among these plots including Figure 4.7 for Feature Set 1, it can be seen that the best clustering is formed using Feature Set 3.

Figures in Appendix A.5 represents clustering formed by PCA on relative variation scaled inverse kinematics dataset and motion quality classified using all the new feature sets. Among these plots including Figure 4.9 for Feature Set 1, it can be seen that the best clustering is formed using Feature Set 3 but still has a high degree of overlap between the clusters.

Figures in Appendix A.6 represent clustering formed by PARAFAC on mean-centered inverse kinematics dataset and motion quality classified using all the new feature sets. Among these plots including Figure 4.12 for Feature Set 1, it is evident that the best clustering is formed using Feature Set 3 in this analysis as well, but since there is a high degree of the mix in the clustering, we do not take PARAFAC along with mean centered dataset for further analysis.

From the analysis using PCA and PARAFAC using all the feature sets, it is found that the Feature Set 3 resembles the body balance well as compared to the other feature sets. It also proves that the muscles between the hip joint and knee joints are responsible for attaining the body balance during the "sit-to-stand" motion and hip joint loads and knee joint loads do not contribute in the attainment of body balance.

Table 4.2 shows the result of feature set selection after analysis on mean-centered and relative variation scaled datasets using PCA and PARAFAC. It is found that the best clustering of patient using motion quality categories if formed using Feature Set 3 by performing PCA. We are interested in selecting only one feature set using either PCA or PARAFAC to get the best results, therefore, any other clustering which is close but not as good as PCA on the Feature Set 3 is not considered for the further analysis.

It can be understood from the analysis that PCA is a better dimensionality reduction technique for the analysis of motion quality than the PARAFAC technique

Mean Centered									
Analysis \ **Feature Set**	FS1	FS2	FS3	FS4	FS5	FS6	FS7	FS8	FS9
PCA	No	No	**Yes**	No	No	No	No	No	No
PARAFAC	No	No	No	No	No	No	No	No	No
Relative Variation Scaled									
Analysis \ **Feature Set**	FS1	FS2	FS3	FS4	FS5	FS6	FS7	FS8	FS9
PCA	No	No	No	No	No	No	No	No	No
PARAFAC	No	No	No	No	No	No	No	No	No

Table 4.2: Selection of feature sets based on PCA and PARAFAC analysis

as PCA works with a 2-way vector whereas PARAFAC works with a 3-way vector. There is a loss of information while remodeling the data in 3 dimensions which affects the motion quality analysis.

Hence, the motion quality quantification is evolved by removing the hip joint loads, knee joint loads and rectus femoris muscle loads from the Feature Set 1, which acts as noise when considering the motion quality analysis based on body balance. Therefore, the most important muscles responsible for the body balance of lower extremity during a "sit-to-stand" motion are vastus medialis, vastus intermedius, and vastus lateralis. Table 4.3 shows the new distribution of patients in the motion quality classes "Fair" and "Poor" quantified using Feature Set 3.

Fair	Poor
23	25

Table 4.3: Distribution of patients in motion quality categories using Feature Set 3

4.8 Input Feature Selection

From the analysis in Section 4.7, we achieved a clustering for further investigation by applying PCA on mean-centered inverse kinematics features. The current input set of inverse kinematics consists of 19 features that represent the joint angles of the lower extremity during a "sit-to-stand" motion cycle. It is not necessary that

all the features are relevant for the study of motion quality that depicts the body balance, hence, selection of most relevant features from the inverse kinematics dataset is performed using the results of the PCA.

From Figure 4, it can be seen that the important principal components out of PC1 and PC2 is PC1 as the cluster separation is achieved by the distribution of data points on the PC1. Also, PC1 consists of the maximum variance of the original 19 inverse kinematics features as compared to the other principal components, hence, we perform the feature selection using the information collected in the PC1, .

As discussed in Section 2.5.1, during the PCA, the original data points are projected to the orthogonal vectors called principal components, and after minimization of the mean squared error of all the projections, the final principal components are generated. The distance between the point of projection on the principal component and the center of principal component, "zero", is called a loading for the original data point. These loadings are responsible for accumulating the variance from the original dataset onto the principal components. The bigger the loading, the more is the relevance of the feature for the principal component. Hence, feature selection is performed by sorting the loading magnitudes and selecting the features with higher magnitudes.

Figure 4.15 shows a bar plot of PCA loadings for 19 inverse kinematics features with the magnitude of loading on the y-axis and the indices of the linearly stacked time series feature on the x-axis. The feature indices with higher loadings are shown above the peaks from which the feature selection is performed. We selected the top 6 features with maximum loadings whose index and feature mapping is shown as follows:

1. Index 437 - Pelvis tilt on y-axis
2. Index 346 - Pelvis tilt on x-axis
3. Index 941 - Knee flexion of right leg
4. Index 1440 - Knee flexion of left leg
5. Index 1088 - Right ankle angle
6. Index 1594 - Left ankle angle

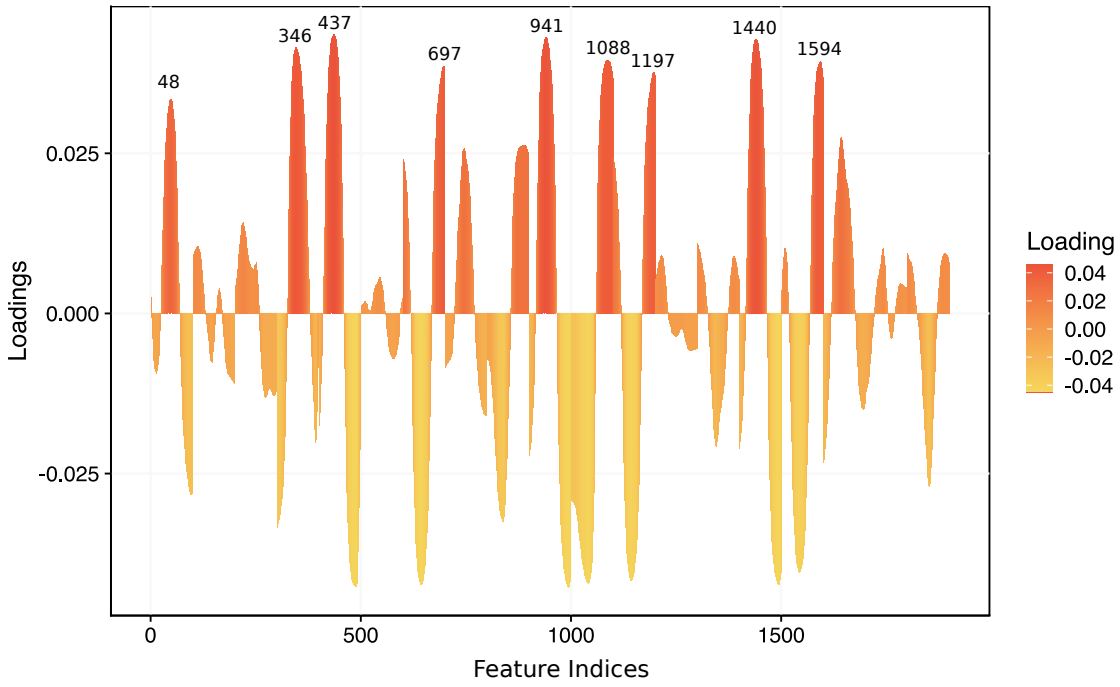


Figure 4.15: PC1 loadings for mean centered inverse kinematics features

It can be seen from the above selection that the top 6 features are in pairs of the same kind of feature on a different axis and both sides of the lower extremity. This is a good result which resembles the quantified motion quality using delta forces on the muscles that depict the balance of the lower extremity of the human body during the "sit-to-stand" motion cycle.

From this analysis, we understand the features like pelvis tilt, knee flexion, and ankle angle resembles the motion quality quantified using the quadriceps femoris muscles. The motion quality can be predicted using ANN models using these selected joint angles.

Figure 4.16 shows PCA results using the top 6 inverse kinematics features, and cluster formation using the Feature Set 3. It has a total of 9 misclassifications in both the clusters when visualized with the PC1 and PC2.

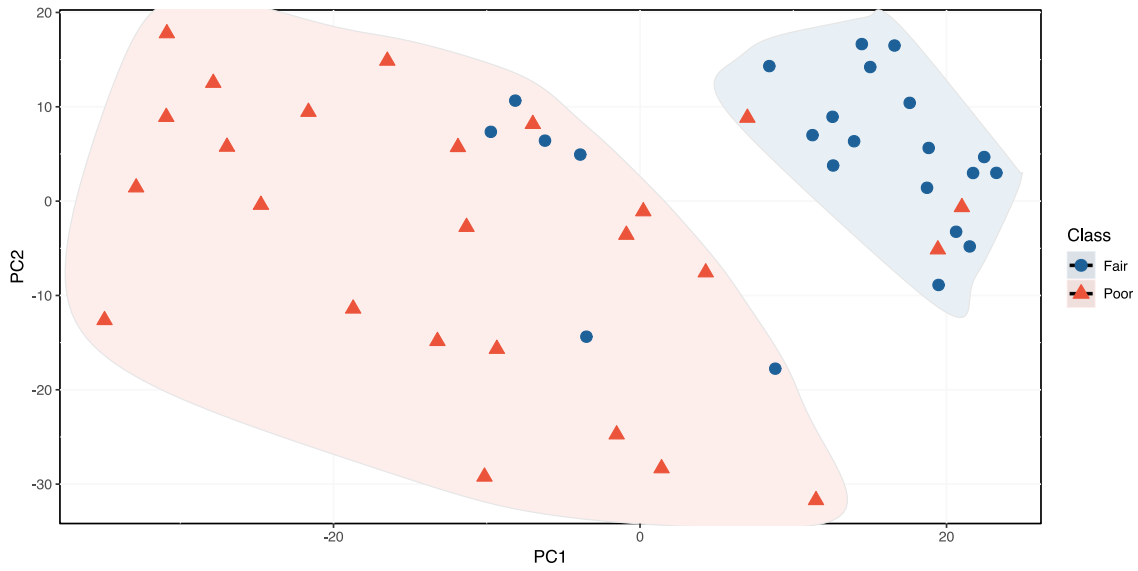


Figure 4.16: Clustering form by PCA using mean centered top 6 inverse kinematics features and motion quality quantified using Feature Set 3

Chapter 5

Research and Analysis - Phase 2

5.1 Introduction

To bridge the gap between the inverse kinematics stage of musculoskeletal data simulation and the motion quality quantification, by eliminating the inverse dynamics and muscle force prediction stages, we utilize ANN by performing supervised learning with the inverse kinematics features as the predictors and the classified motion quality as the predictions. The ANN training is conducted using such a technique that, even though the training is performed using a small number of records, it can efficiently predict the motion quality from any unseen dataset. For this purpose, we use some of the muscle forces and joint angles of the lower extremity that are the result of motion quality evolution and input feature selection, as discussed in Chapter 3.

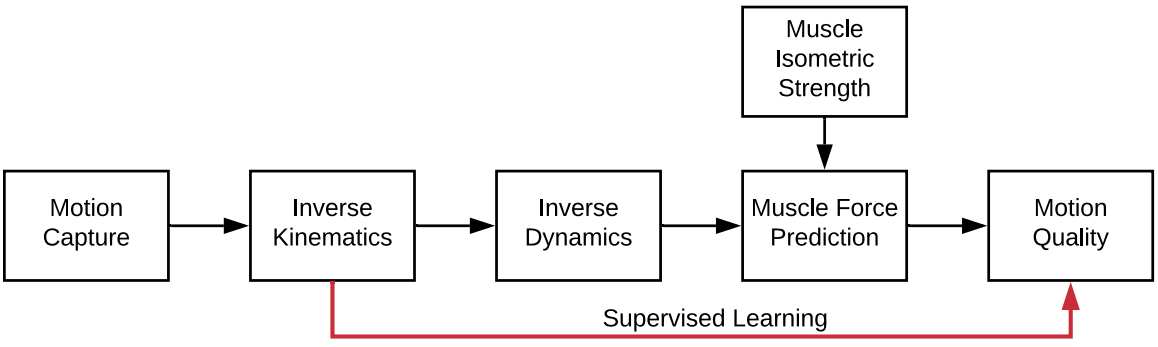


Figure 5.1: Bridging the gap between inverse kinematics and motion quality

Apart from deploying multiple variants of ANN for the prediction of motion quality, we also attempt to contrast a conventional regression technique - Logistic regression with ANN in terms of their performance on the same dataset.

The outcome of this section explains how we can approach to improve the ANN models like feedforward, CNN, and LSTM that uses the wide-shaped dataset for supervised learning. Moreover, it also explores experimental techniques like autoencoders and merged networks of different ANN variants like CNN and LSTM for efficient supervised learning.

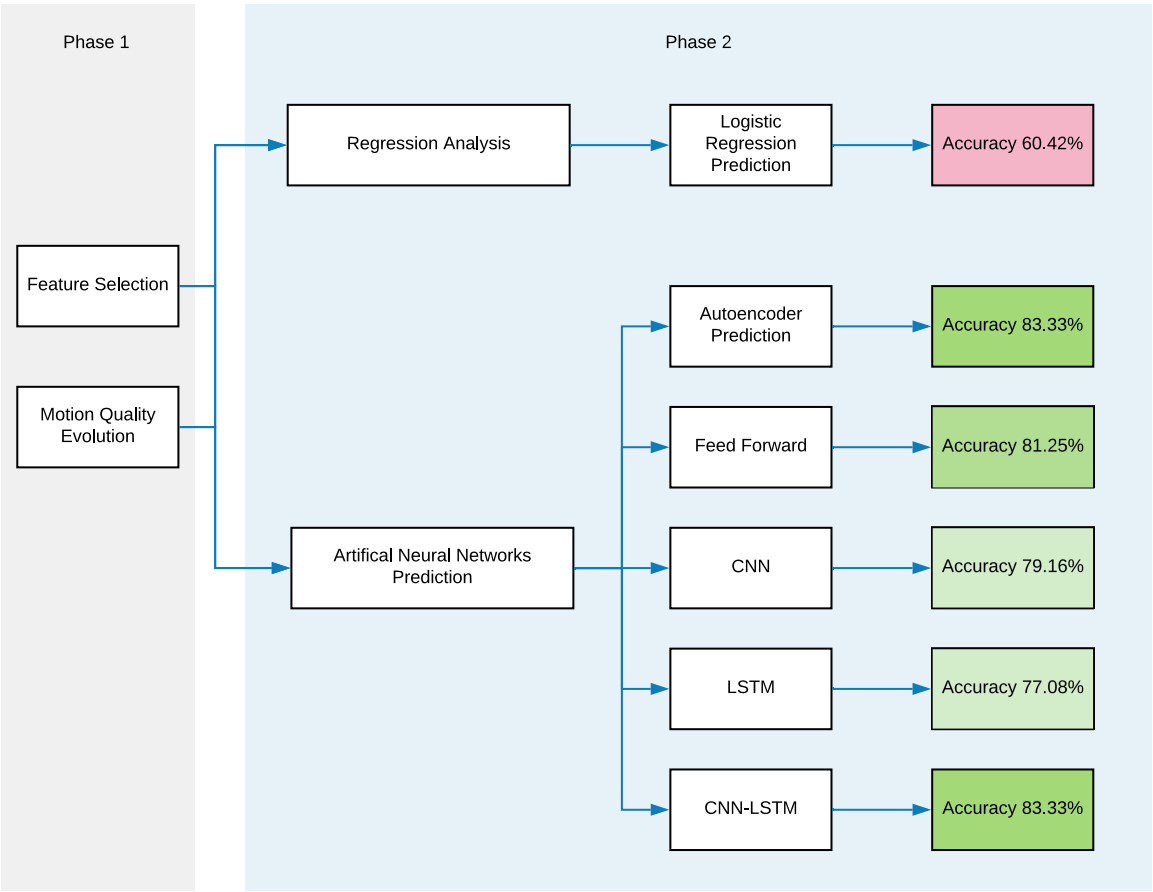


Figure 5.2: Research and Analysis, Phase 2, work flow

Figure 5.2 describes the flow of research and analysis we perform using multiple machine learning techniques to predict the motion quality of THA patients using the refined input dataset and motion quality classified in Phase 1.

5.2 Logistic Regression - Motion Quality Prediction

Logistic regression is one of the conventional regression method for binary classification. As the motion quality is divided into 2 categories: "Fair" and "Poor", and the dataset contains 600 columns per row, logistic regression is a suitable regression technique to use for the binary classification using multivariate dataset.

For the classification, we trained the logistic regression model for 48 iterations leaving out one patient from the training dataset to be used as testing dataset. This is how we can effectively analyze the performance of the regression on the unseen joint angle dataset.

The combined accuracy produced by 48 regression iterations for the correct motion quality prediction is 60.42% by misclassifying 29 patients. Hence, multi-variate conventional regression analysis are unable to analyze the dataset for an effective motion quality prediction, therefore we try to improve the prediction using ANN in the further sections to contrast the effectiveness of ANN over regression techniques.

5.3 ANN - Motion Quality Prediction

The classification of patients' motion quality was performed using the muscle forces of the lower extremity. The muscle force prediction stage is dependent on the second stage inverse dynamics, which was used to calculate the forces on joints during a motion cycle. This simulation process is time-consuming, which can take weeks to simulate the data to classify a patient's motion quality.

To bridge the gap between motion quality and inverse kinematics, ANN are used that are trained using 6 inverse kinematics time-series features and classified motion quality using Feature Set 3. Different kinds of networks are used to predict patients' motion quality which addresses different ML approaches that can be used for many types of application.

For training a neural network, input dataset requires segregation into three different sets, namely: training data, validation data, and testing data.

1. **Training dataset:**

The training dataset is a part of the original dataset which is used to train the neural network. The classification of patients in the training dataset is used to verify the learning. According to the result of the network's output and the actual motion quality, the neural network performs back propagation, implicitly, to tune the network by adjusting the weights of synaptic connections between the neurons.

2. **Validation dataset:**

   The validation dataset is a part of the original dataset which is used to validate the learning of neural network by predicting the category of motion quality of the patients in the validation dataset after the network is trained using a training dataset. During the learning iterations, called epochs, the validation dataset is used to understand about network's performance on unseen data. It helps in deciding the number of times a neural network should be trained to predict the quality of motion of unseen data correctly.

3. **Testing dataset:**

   The testing dataset is a part of the original dataset which is used to test the learning of the network based on the training dataset. This unseen data is used to report the final accuracy of the network's prediction. The accuracy is the measure of correct predictions reported as a percentage of total data points in the testing dataset.

The ANN training requires the right balance between the number of features and the number of data points, where a data point refers to a patient for this research. As we use six time-series with 100 points each, the neural network assumes that 600 features have been provided for 48 data points for learning. This kind of dataset is called a wide dataset, which has a minimal number of rows as compared to the total number of columns. In the AI research, this is an interesting problem as conventionally the neural networks trains over a large dataset consisting of more rows than columns. Moreover, the dataset of THA patients' "sit-to-stand" motion is even more complicated being a set of deterministic time-series features. For such a dataset, a technique called cross-validation has been used to train and test the neural networks.

5.3.1 Cross Validation:

During the training of the ANN, it is essential that the network understands most of the information embedded in the dataset pertaining to the prediction of motion quality. Instead of dividing the dataset in training, validation and testing subsets, 48 iterations are performed, and in each iteration, one patient is left out from the training dataset and is used to test the network's training by acting as a testing dataset with only one row. Hence, we do not use validation dataset. The accuracy reported will be the percentage of the total number of correct predictions of the category of motion quality among 48 patients. This approach is called as cross-validation and is useful for the ANN training with wide datasets.

5.3.2 Undercomplete Autoencoder

Undercomplete autoencoder is an ANN dimensionality reduction technique that is inspired by the idea of principal component analysis. As discussed in Section 2.9.4, neural networks can learn non-linear relationships. Therefore, the ANN autoencoder is a better dimensionality reduction technique as compared to the linear techniques; PCA and PARAFAC.

For constructing an autoencoder network, the first step is to decide the number of neurons in the encoding layer as the number of data points in the encoded format will be equal to the number of neurons in the encoding layer. As discussed in Section 2.5.1, the maximum number of components that can be extracted using PCA is 48 according to the shape of the dataset in the application. Therefore, we decided the closest rounding of 48, which is 50 neurons in the encoding layer.

The architecture of an undercomplete autoencoder requires to have a decreasing number of neurons in the subsequent layers in the encoder part of the network and mirroring the same, the decoder part of the network requires to have an increasing number of neurons in the subsequent layers. Following this approach, we construct the autoencoder network as depicted in Figure 5.3 for dimensionality reduction.

There is no standard rule to decide the number of neurons in the layers other than the encoding layer. Therefore, we decided to at least half the number of neurons in each subsequent layer of the encoder part and at least doubled in the successive layers of the decoder part.

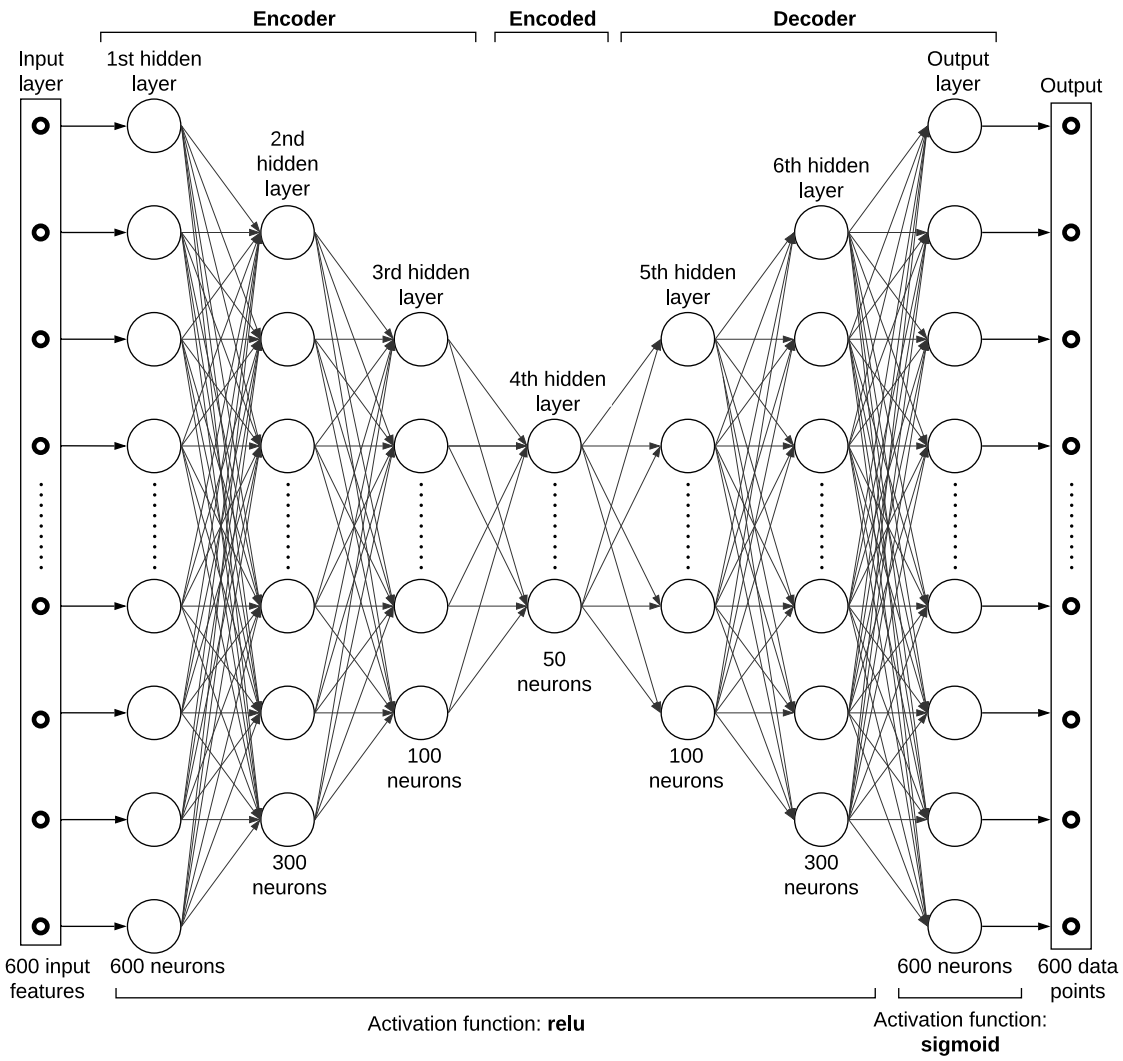


Figure 5.3: Undercomplete autoencoder used for dimensionality and noise reduction on inverse kinematics dataset

Data Remodeling: The input features in the Feature Set 3 have different ranges of magnitudes as shown in Figure 5.4, due to which the importance of values of each time-series feature will not be consistent. The overall variance of the feature series is 229.83. To use the features in the autoencoder, we remodel the dataset by shrinking the overall range of the features between 0 and 1. Figure 5.5 depicts the resultant feature set, which has a reduced variance of 0.033. It can be seen from the figure that the overall pattern of the features is similar to the original dataset and persists their original information.

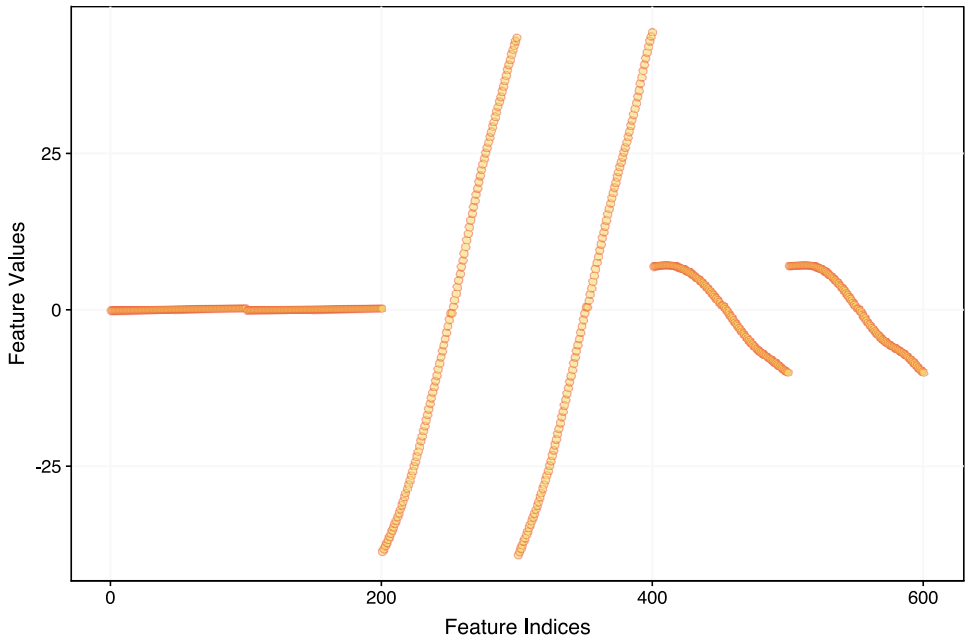


Figure 5.4: Mean centered feature set 3

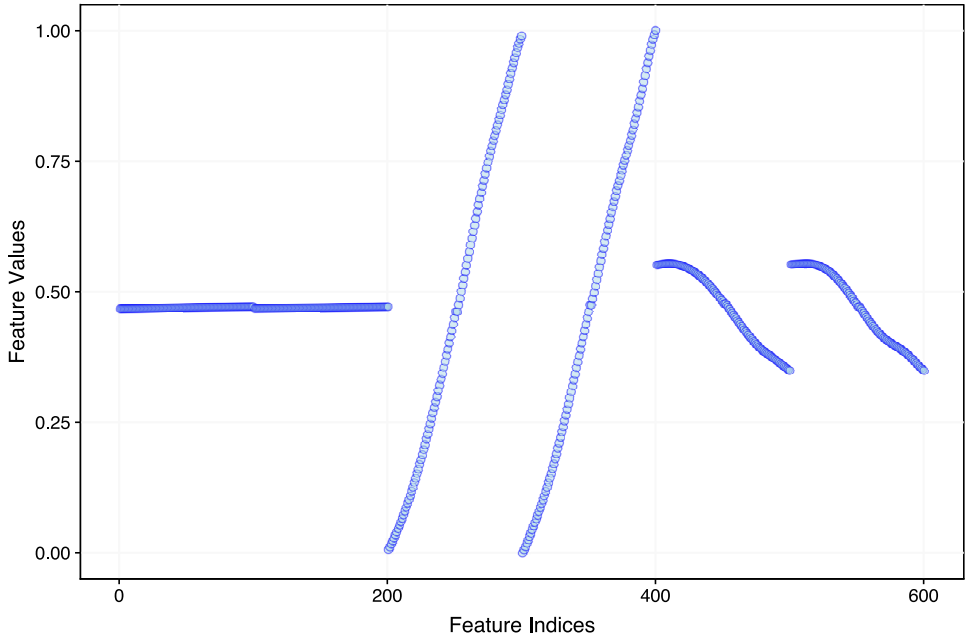


Figure 5.5: Feature set 3, Remodeled, Range 0 to 1

Since the values of the transformed features are in the positive domain, we use ReLu activation function for all the layers which provide the actual value of the

input dataset to each neuron. For the output layer, we use the sigmoid activation function, which is proven effective for the reconstruction of the dataset as compared to ReLu after performing multiple iterations using different combinations of activation functions. Since an autoencoder is expected to work similar to the PCA, we use minimum squared error as the loss function. The training of the network is performed for 10,000 epochs using Adam optimizing algorithm.

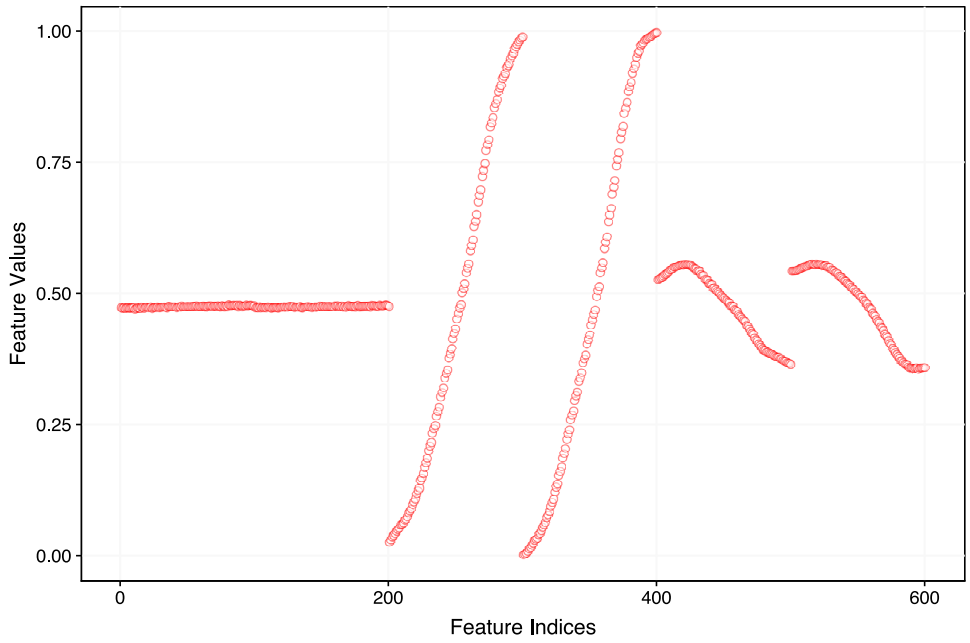


Figure 5.6: Reconstructed input dataset using the autoencoder

Figure 5.6 shows the output of the autoencoder, which is almost similar to the remodeled input dataset. There is some deviation in the reconstructed output from the remodeled input, which depicts the reduction of noise during the reproduction. The output from the encoded layer consists of 50 data points for each patient. Therefore, we achieve dimensionality reduction from 600 to 50 attributes, which is 91.67% compression of the original information.

The compressed data for patient 1 extracted from the autoencoder is shown in Figure 5.7. From the figure, it can be seen that the data points are scattered with no visually comprehensible information, and it is the expected behavior because it is a representation of encoded information. It is also evident from the figure that not all the neurons are active in the encoding layer for the reconstruction of the feature set

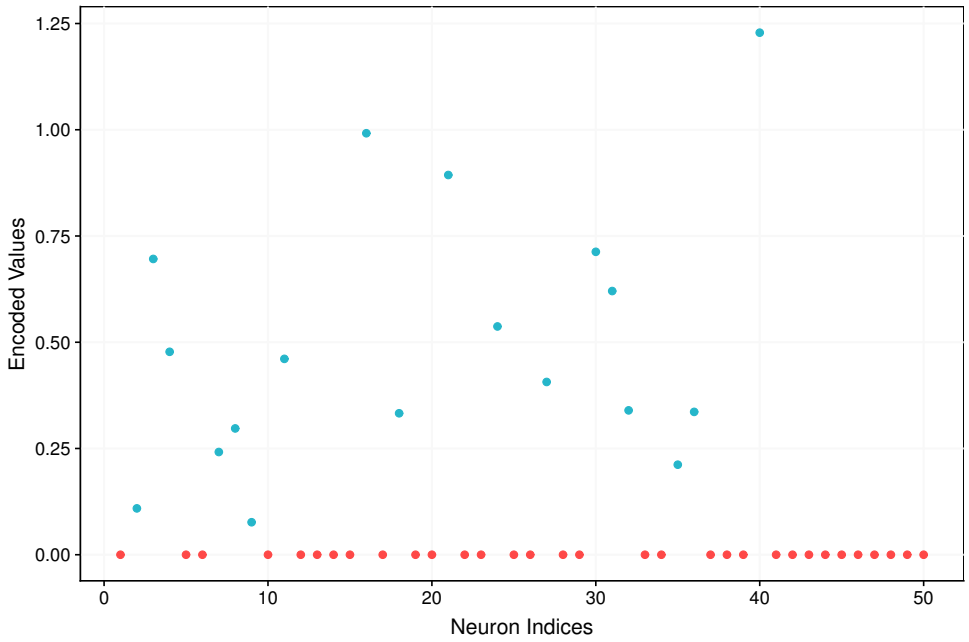


Figure 5.7: Output of the encoding layer for patient 1

for patient 1, depicted by zero value points. There are different sets of active neurons in the encoding layer for each patient that provides an input to the decoder part of the network, which includes necessary information for the feature reconstruction.

The encoded information is extracted from the encoding layer, which is further used as an input to a feedforward neural network for the prediction of motion quality of each patient using the cross-validation technique.

The network for motion quality prediction is developed using the concept of the bias-variance dilemma, discussed in Appendix A.1, by starting with a minimum number of neurons and layers in the beginning. After training the feedforward models, and according to the accuracy received, the network is grown by adding more neurons and layers. We have used Adam optimizing algorithm with a categorical cross-entropy function as the loss function for the network's training and 600 epochs for each cross-validation iteration.

The prediction starts with the smallest network that is shown in Figure 5.8, which has one hidden layer with two neurons and an output layer with two neurons. The two neurons in the output layer will have a probability score for the prediction of the two classes: "Fair" or "Poor", respectively. By performing cross-validation training

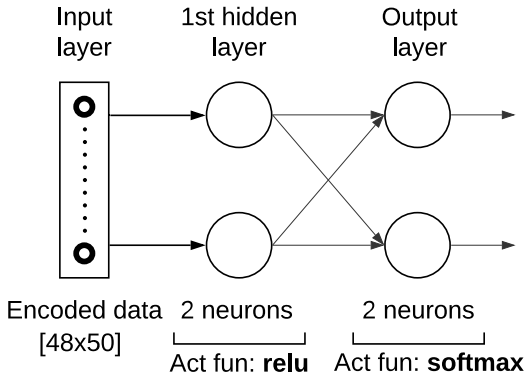


Figure 5.8: Feedforward followed by autoencoder, iteration 1, 64.58% accuracy

on this network, we get an accuracy of 64.58%. For this accuracy, the network has misclassified a total of 17 patients. According to the bias-variance tradeoff of the network, the network has high bias, and a low variance as the network's complexity is low because of only two neurons in the hidden part. From this, we understand that increasing the network's complexity can give better results.

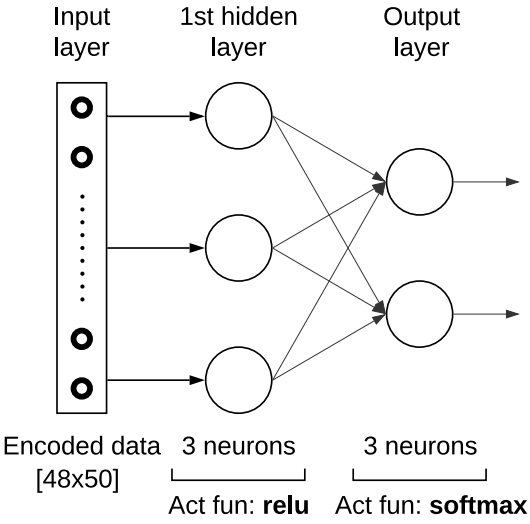


Figure 5.9: Feedforward followed by autoencoder, iteration 2, 70.83% accuracy

Figure 5.9 represents the network used in the second iteration of the process of network improvement, which has one hidden layer with three neurons. Performing cross-validation training on this network, we get an accuracy of 70.83%. For this accuracy, the network has misclassified a total of 14 patients.

This network performed better as compared to the previous network by adding one neuron in the hidden layer. This demonstrates the increasing variance and decreasing bias in the network. The result from this network is still not as good as the clustering we observed by PCA on the 6 inverse kinematics features as represented in Figure 4.16, which has a total of 9 misclassifications. As the encoded information has high variability, for further analysis, we aim to add more layers and neurons to increase the variance of the network.

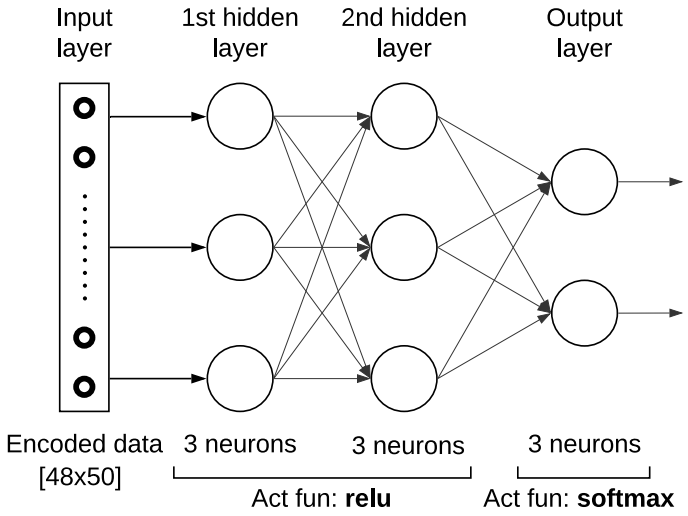


Figure 5.10: Feedforward followed by autoencoder, iteration 3, 75% accuracy

Figure 5.10 represents the network used in the third iteration, which has two hidden layers with three neurons each. Performing cross-validation training on this network, we get an accuracy of 75.0%. For this accuracy, the network has misclassified 12 patients.

This network performed better as compared to the previous network by adding one layer with three neurons in the second hidden layer. This network too, demonstrates the increasing variance and decreasing bias in the network. From this attempt, we understand that increasing the complexity of the network produces a better accuracy, and we aim to perform more iterations to achieve the desired prediction of motion quality similar to that of PCA.

Figure 5.11 represents the network used in the third iteration, which has three hidden layers with three neurons each. Performing cross-validation training on this

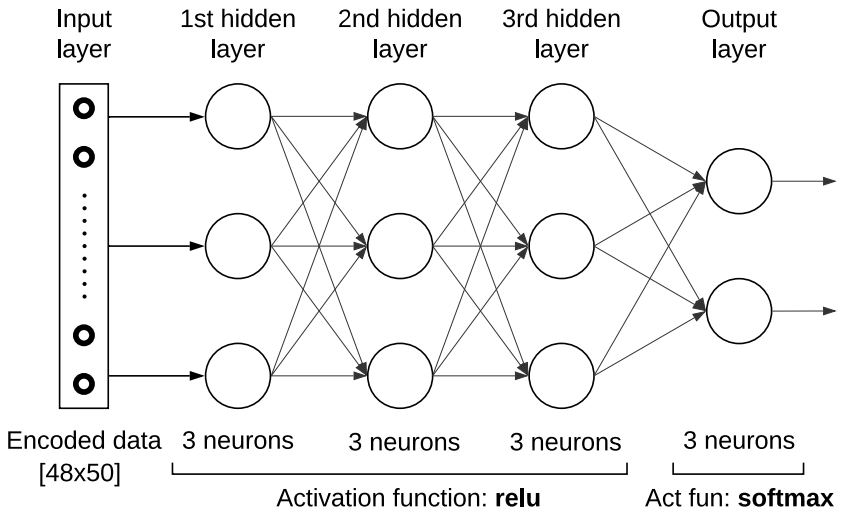


Figure 5.11: Feedforward followed by autoencoder, iteration 4, 70.83% accuracy

network, we get an accuracy of 70.83%. For this accuracy, the network has misclassified 14 patients.

This network does not perform better than the previous network by adding one layer with three neurons as the third hidden layer. This is a case of bigger increase in the bias than an increase in the variance of the network. Therefore, this network demonstrates increasing bias and decreasing variance in the network that introduces errors which leads to more misclassifications. From this attempt, it is understood that increasing the number of layers in the network will need to introduce more neurons to add more variance to achieve the desired accuracy.

After performing numerous iterations, Figure 5.12 represents the network used, which is the most effective so far. In this iteration of the process of network improvement, we add one additional neuron in the three hidden layers, respectively, as compared to the network in Figure 5.11. The loss over 48 cross-validation iterations is depicted in Figure 5.13 which shows that on an average, the loss is continuously decreasing after each epoch for all 48 iterations. By performing training on this network, we get an accuracy of 83.33% for motion quality prediction. For this accuracy, the network has misclassified a total of eight patients. This network has performed better classification as compared to the PCA analysis shown in Figure 4.16, which has nine misclassifications.

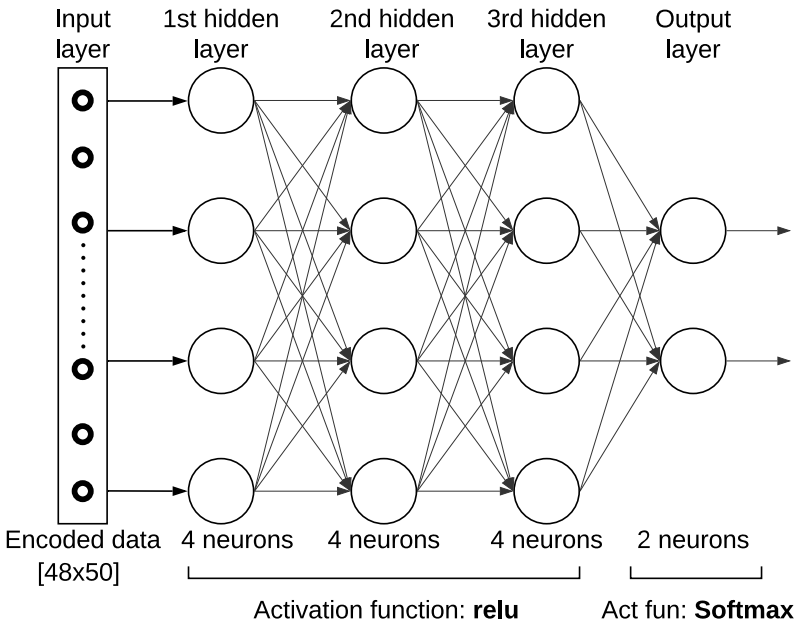


Figure 5.12: Feedforward followed by Autoencoder, final iteration, 83.33% accuracy

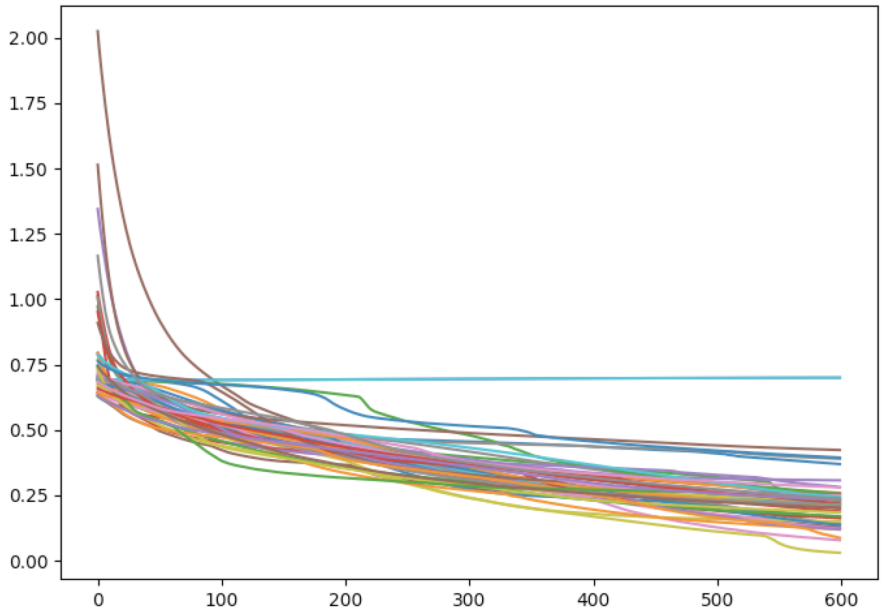


Figure 5.13: Loss curve of the 48 cross-validation training steps using network in the final iteration

From some of the other iterations, we find that increasing the network's complexity also increases in the variances, which leads to overfitting of the model. Hence, it

produces a lower accuracy prediction. Therefore from this attempt, we understand that having the right balance between variance and bias of a network, we can expect better results from a neural network than that of the component analysis techniques.

If it was possible to visualize the results of the network in multi-dimensional space, we could see that the actual clustering of the data points generated by the computation of the network on encoded information has eight misclassifications. Hence, 83.33% accuracy is our desired accuracy, and we can conclude that a combination of an autoencoder and a feedforward network can effectively predict the motion quality.

The ANN prediction attempt using a combination of an autoencoder and a feedforward network is successfully able to classify the patients in the classes: "Fair" and "Poor". This analysis has demonstrated that the autoencoder can compress the most relevant information from the original dataset in 50 data points for each patient, which then used as an input to the feedforward network produces the desired results.

5.3.3 Feedforward Neural Network:

The feedforward networks consist of elementary neurons and are the most basic type of neural networks. The network is tuned to predict the correct motion quality using back-propagation. We develop the feedforward network by improving the bias-variance tradeoff as explained in the section 5.3.2. For the prediction, we use mean centered selected features of joint angles, therefore the input dataset has size 48x600 with 6 time series features having 100 time units each. For the network's training and testing, we use cross-validation technique.

To start with a basic network, Figure 5.14 represents the feedforward neural network that we use to begin the prediction of the motion quality. As the input dataset is bigger as compared to the encoded dataset in Section 5.3.2, for the most straightforward network to start with, we consider having one layer with five neurons. The activation function used for the neurons in the hidden layer is tanh as it returns appropriate values for the whole range of numbers and scales them down in the range of -1 to 1. We use the softmax activation function for the output layer as it returns the probabilities for each class and is suitable for multi-class classification. We use categorical-cross-entropy as the loss function and Adam optimizing algorithm to optimize the leaning of the network.

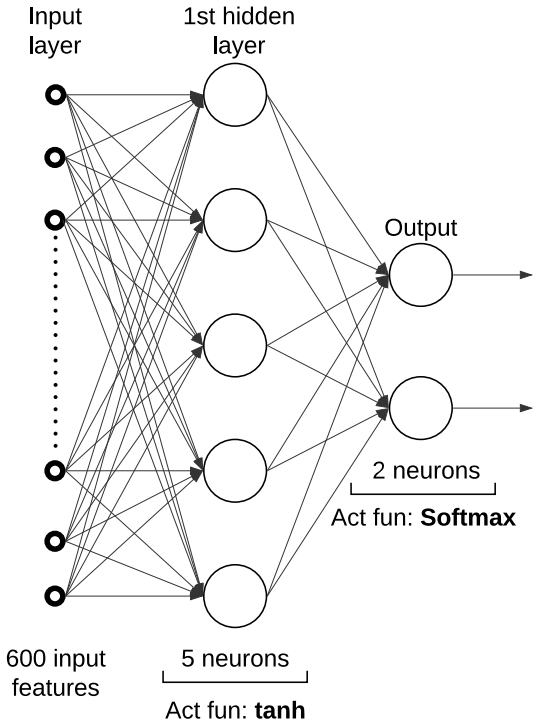


Figure 5.14: Feedforward, iteration 1, 75.0% accuracy

The network in Figure 5.14 has produced an accuracy of 75.0% and has misclassified 12 patients. As the dataset used has high variance, we aim to improve the network by adding more neurons to the first layer to have a higher variance in the network's unknown function.

The network in Figure 5.15 is developed by adding 5 more neurons in the first hidden layer, and it has produced an accuracy of 75.0%. This result suggests that the change in bias and variance is uniform as compared to the last network used. We aim to improve the network by adding an additional layer to add more computation and higher variance in the network's unknown function.

The network in Figure 5.16 is developed by adding one more hidden layer with ten neurons, and it has produced an accuracy of 77.0% by misclassifying 11 patients. This result demonstrates an increase in variance and a decrease in the bias of the network's unknown function. From this result, we understand that we should increase the variance of the network by introducing more layers to achieve the desired accuracy.

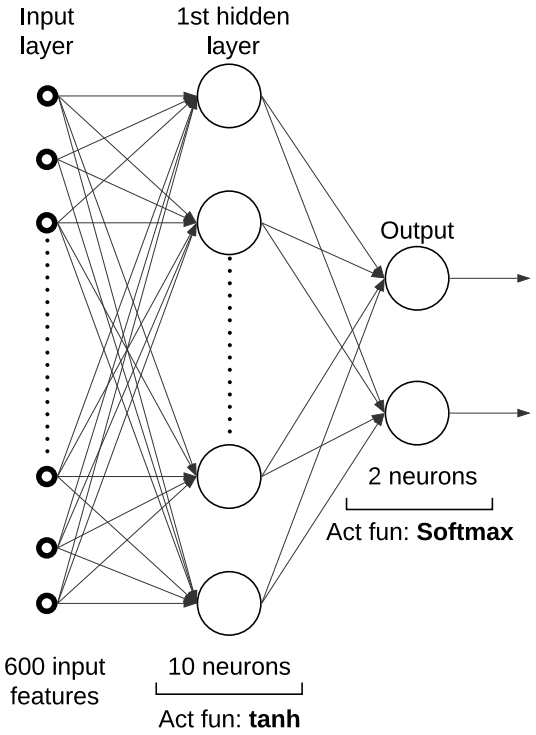


Figure 5.15: Feedforward, iteration 2, 75% accuracy

After numerous iterations with different networks, the network in Figure 5.17 produced the highest accuracy of 81.25% by misclassifying 9 patients. The network has a total of three hidden layers with 20, 5, and 3 neurons respectively, and is converging towards to output layer as the subsequent layers have a reduced number of neurons. This network represents that the information that is required to predict the motion quality gets concentrated and filtered towards the output layer.

The feedforward neural network demonstrates a successful multi-class classifier application for THA patients by predicting their motion quality with an accuracy of 81.25% by misclassifying nine patients. The network can classify the motion quality using the mean centered time series data as an input and produces the same classification as it is shown in the clusters generated by the component analysis technique represented in Figure 4.16. The clusters in the figure also have nine misclassifications, and the feedforward network can imitate the same results.

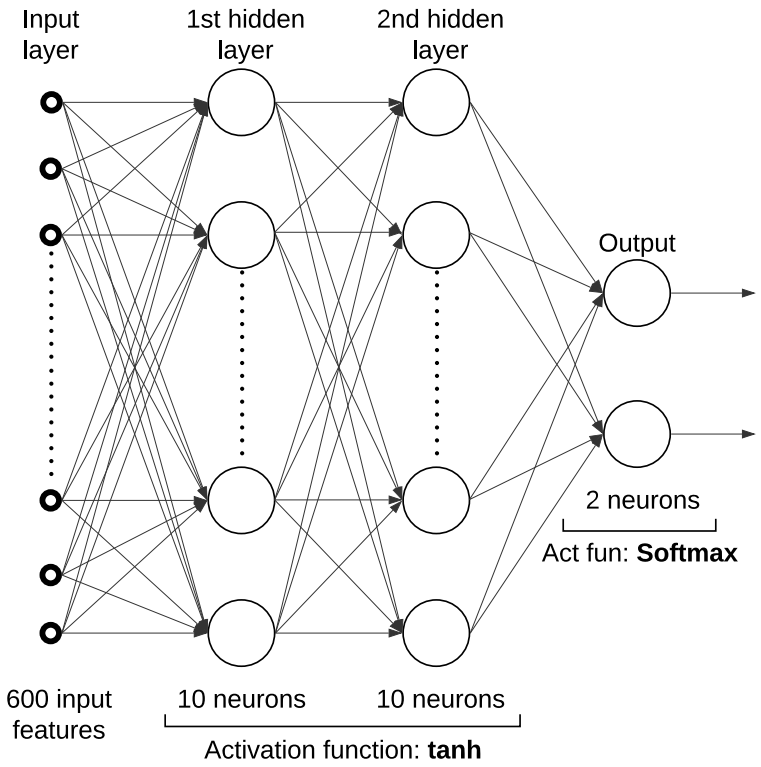


Figure 5.16: Feedforward, iteration 3, 77% accuracy

5.3.4 Convolutional Neural Network:

Another variation of ANN we use is the convolutional neural network to classify the patient's quality of motion by analyzing the 6 joint angle features as a heatmap. The heatmap is provided to the network as an input image. The input image is a matrix of numerical time series values of size 6 rows and 100 columns. The size of the kernel used for the input layer is 3x3, that is, the image will be broken into matrices of size 3x3, and that becomes a granular feature of the image that will be taken as an input for the next layers for classification of motion quality.

Figure 5.19 represents the CNN used in the first iteration of the network construction process. It convolves a 3x3 kernel over the input image that creates a new image of size 4 rows and 98 columns consisting of granular features of the original input image. There is only one filter used in the first convolutional layer, and the new image becomes the output of the filter. In the next step, max pooling is performed using one filter which uses the pool size of 2x2 on the output of the convolutional layer.

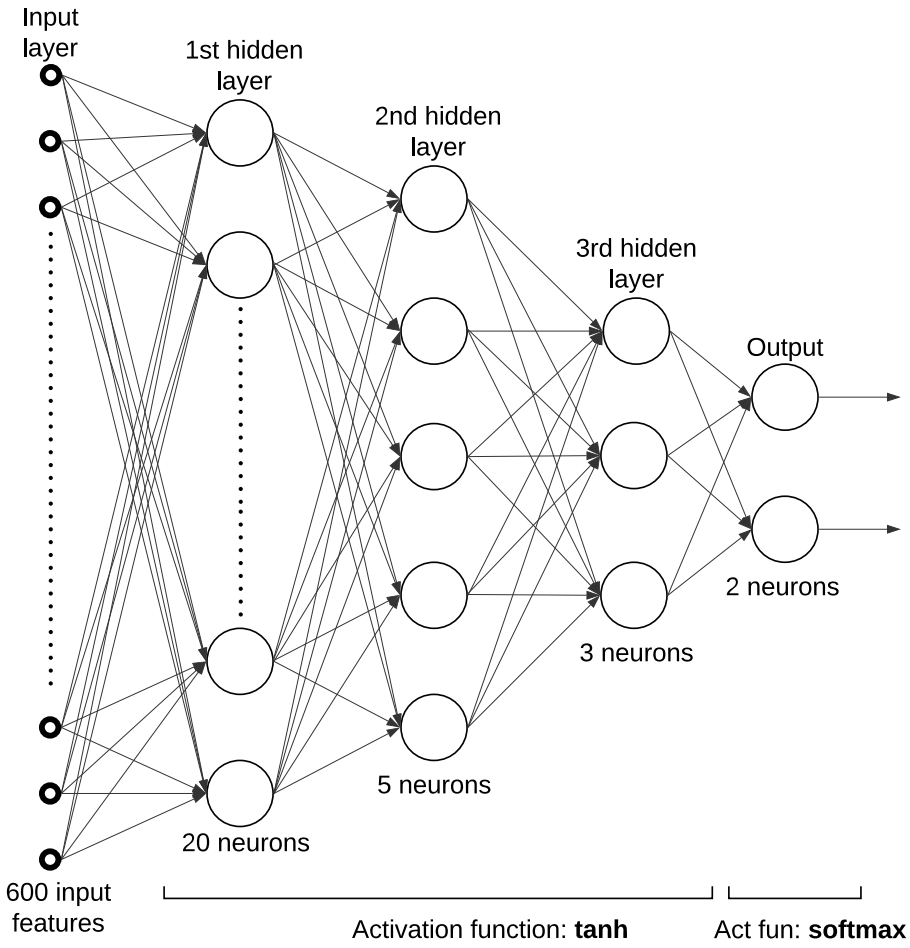


Figure 5.17: Feedforward, final iteration, 81.25 % accuracy

The max pooling will fetch a segment of the second image of size 2x2, and the result of pooling will be the maximum value in the 2x2 image segment. Hence, pooling minimizes the granular image structures by 75%, and this information is relevant for the motion quality prediction. The size of the resulting image has two rows and 49 columns. This is the process of minimizing the size of the input image based on the highest magnitude values by considering the most significant features in the input image.

The next layer is a dropout layer which drops out the random 50% values and makes them to zero. The randomly selected 50% neurons do not contribute in the back-propagation and due to this, the network can avoid the situation of overfitting of the model.

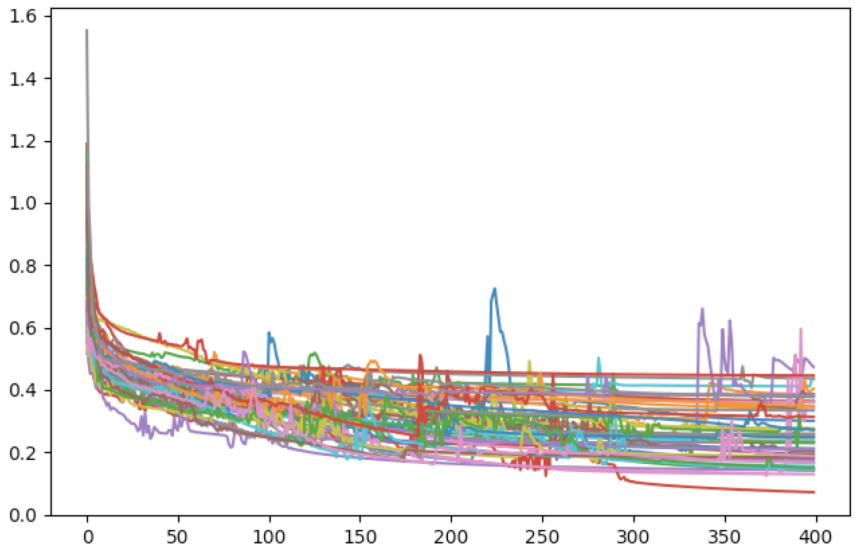


Figure 5.18: Loss curve of the 48 cross-validation training steps using the feedforward network in the final iteration

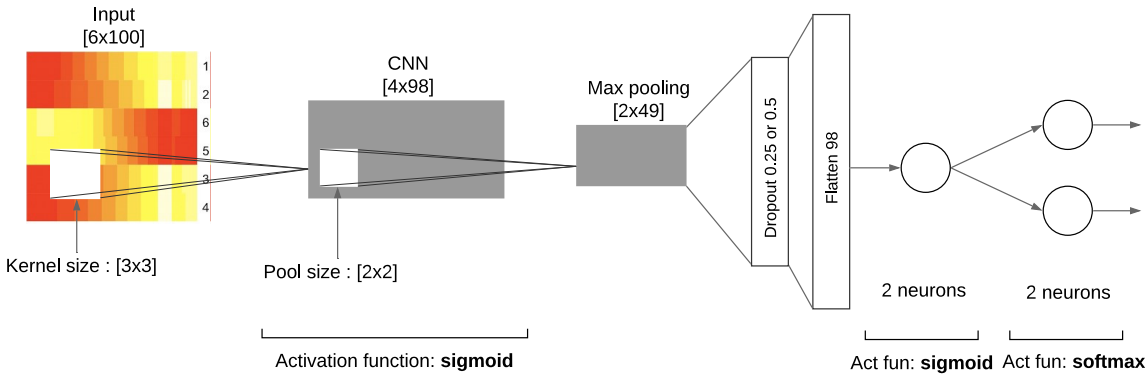


Figure 5.19: Convolutional neural network, iteration 1, accuracy 58.33%

The resulting image is flattened back to a vector of shape 1 x 2 x 49 = 98. This vector is then used as an input to an elementary neuron layer consisting of 1 neuron. The output of the neuron is then forwarded to the 2 elementary neurons, which are the classifiers of this network. The neuron, which has the highest value, becomes the predicted category of the quality of motion for a patient.

The loss function used for the network is categorical-cross-entropy and the optimizing algorithm used is AdaDelta. The activation functions used are sigmoid in the hidden layers, and, softmax, for the output layer. The training was performed

using a cross-validation technique wherein each iteration the network is trained for 100 epochs.

The accuracy produced from this network is 58.33% that misclassifies 20 patients. We tried with a dropout of 25%, and it produced the same result. This is a low accuracy as compared to the previous networks' predictions. The reason for such a low accuracy is using only one CNN unit as it is not able to extract all the critical small parts of the input image. To improve the network we will add more CNN units and experiment with the dropout percentage.

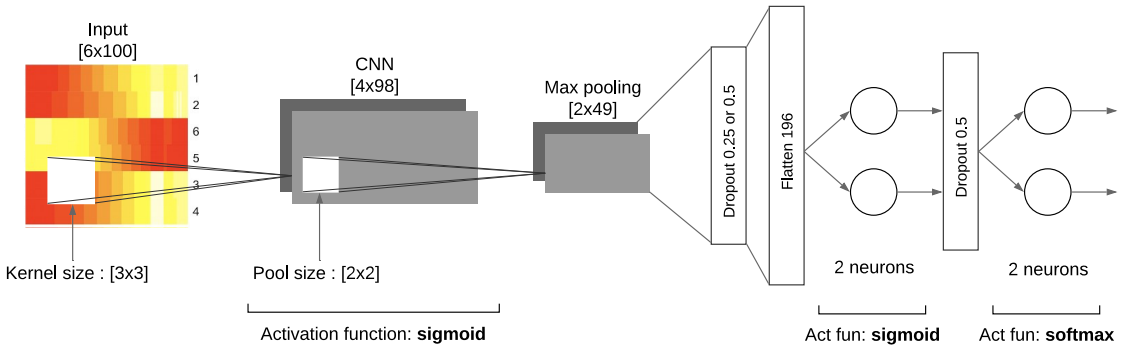


Figure 5.20: Convolutional neural network, iteration 2, accuracy 72.91%

Figure 5.20 represents the CNN network used in the second iteration, which has one added CNN unit, two neurons in the feedforward part following a 50% dropout layer and 25% or 50% dropout for the max pooling layer. This network produces an accuracy of 72.91% by misclassifying 13 patients.

This network's training demonstrated that adding a convolutional layer increases the number of small segments projected to the CNN units that contribute to the information required for the classification. Also, the the size of the input for the feedforward part increases from 98 to 196 values; therefore, more information is passed through the network. The last dropout layer removes the output from any randomly selected neuron out of the two neurons, that helps the network to predict motion quality by eliminating the possibility of overfitting. For further analysis, we add more CNN units for more information flow towards the output layer.

Figure 5.21 presents the CNN network used in the third iteration, which has 3 CNN units, two neurons in the feedforward part following the output layer. There are no dropout layers; hence, there is complete information flow from the input to

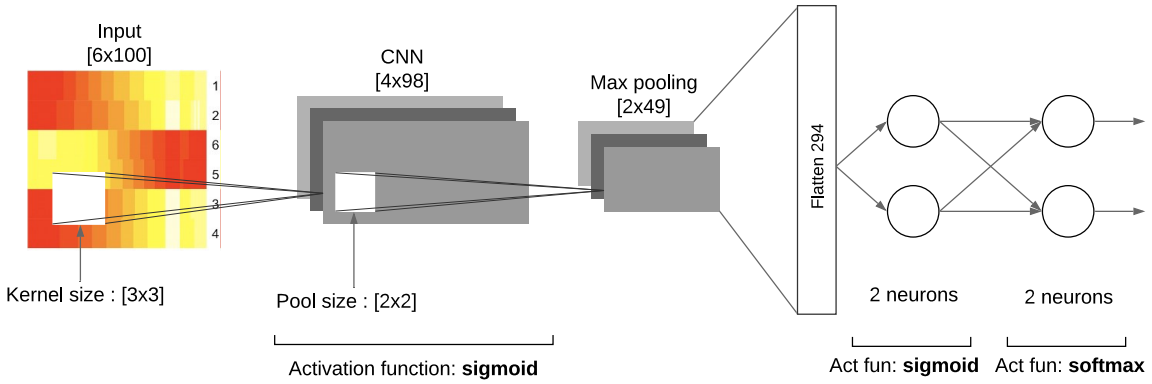


Figure 5.21: Convolutional neural network, iteration 3, accuracy 72.91%

the output layer and every part of the network participates in the back-propagation of errors.

This network produces an accuracy of 72.91% by misclassifying 13 patients. This network's training demonstrated that adding a convolutional layer benefits the network's training, but because of no dropout layers, the prediction did not improve. The network is possibly overfitted to the training dataset. Also, the size of the input for the feedforward part increases from 196 to 294 values, therefore adding dropouts will reduce the noise flowing in the network that can improve the prediction.

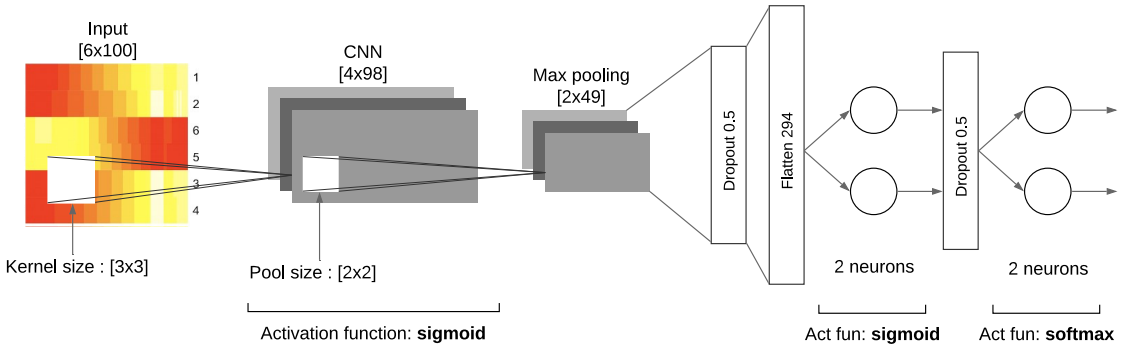


Figure 5.22: Convolutional Neural Network, final iteration, accuracy 79.16%

After trying multiple networks with a different number of CNN units, different dropout layers and feedforward networks, Figure 5.22 produced an accuracy of 79.16% by misclassifying 10 patients. The loss curves for 48 cross-validation iterations are shown in Figure 5.23, which are fluctuating but on an average, it decreases for all the iterations. In this network, we add a 50% dropout after the max pooling layer and

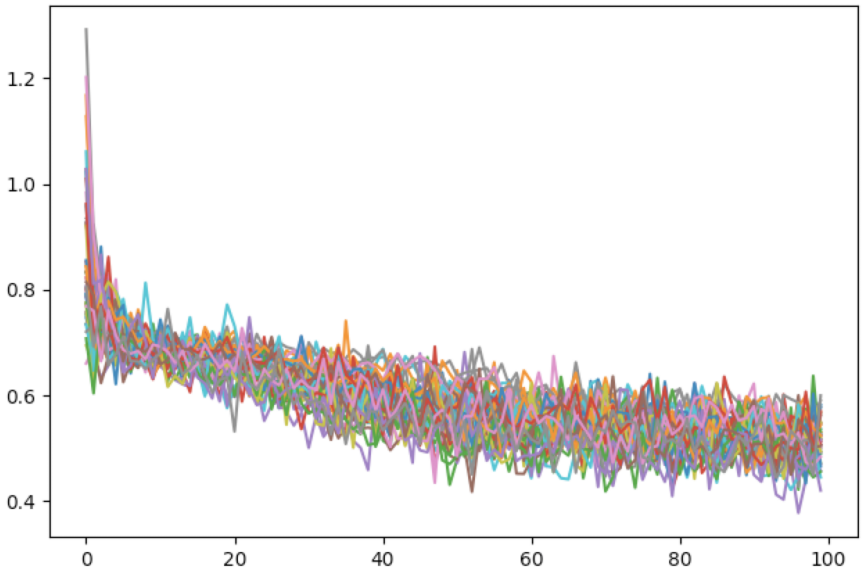


Figure 5.23: Loss curve of the 48 cross-validation training steps using the CNN network in the final iteration

50% dropout after the feedforward two-neuron layer, which eliminates the insignificant information flowing in the network, resulting to a better performing network.

CNN are successful in image classification and object recognition. In this research, we demonstrate that CNN can be successful in identifying the information embedded in the heatmaps of patients' inverse kinematic dataset with time-series values, for the classification of motion quality.

5.3.5 Long Short-Term Memory

LSTM networks are a type of recurrent neural networks that are modeled to work with time-series data that predicts the futuristic values of the input time-series information. In this research, we use LSTM as a multi-class classifier to predict the motion quality of the patients by analyzing the joint angle time-series dataset.

All the six time-series for each patient is provided to the network as an input, separately, so that the network learns about each patient's joint angles at different intervals of training. Therefore, we remodel the input dataset to a 3d vector with shape [48 (patients), 6 (features), 100 (time units)] so that at a time, one slice of the

3d vector containing 6 features with 100 time units will be provided to the network for learning.

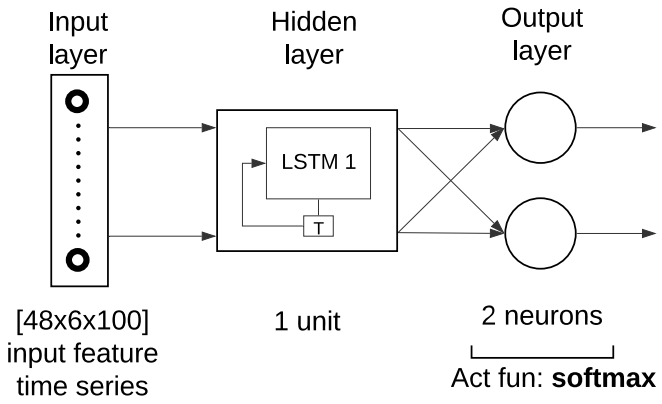


Figure 5.24: LSTM, iteration 1, accuracy 60.41%

We start with a basic LSTM multi-class classifier network shown in Figure 5.24 which consists of only one LSTM unit that follows two elementary neurons for calculating the probabilities of each patient's motion quality. The LSTM unit uses 'tanh' as the activation function and 'hard sigmoid' as the recurrent activation function. The output of this unit contains only one value which is used for the classification by the following layer that uses 'softmax' activation function.

This network produces an accuracy of 60.41% by misclassifying 19 patients. The performance of the network can be improved by adding more LSTM units to the hidden layer to increase the complexity of the network that can analyze the input data more efficiently. More LSTM units will have more recurrent information in the network that can improve the prediction.

In the second iteration, we try with a network shown in Figure 5.25 that has 6 LSTM units in the hidden layer. As the input contains 6 time series features, with this network, we try to emphasize every time-series feature using the equal number of LSTM units. This network produces an accuracy of 72.91% by misclassifying 13 patients. Each LSTM unit produces a single value output for the next layer, therefore for the output layer, the input is a vector with 6 data points.

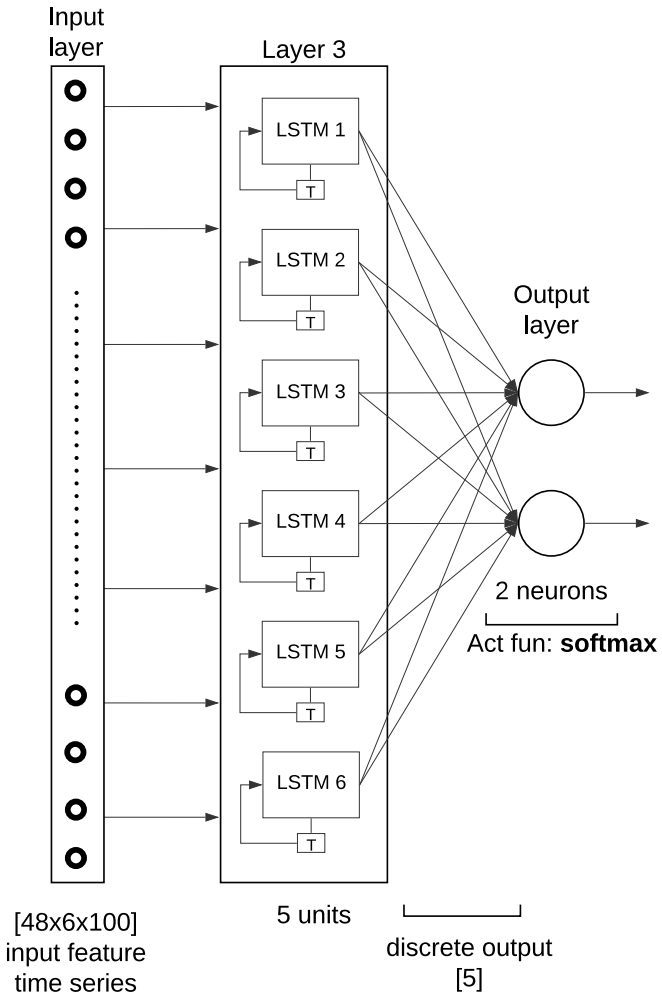


Figure 5.25: LSTM, iteration 2, accuracy 72.91%

To improve the accuracy of the network, we aim at improving the recurrence of the time-series information in the network by adding more hidden layers with LSTM units so that the output of each unit can be a time-series instead of discrete values.

The network used in the third iteration is shown in Figure 5.26 that is developed by adding one layer with 6 LSTM units. The output of the first hidden layer consists of 6 time series with 100 time units. This layer helps in reducing noise and reproduces a new time series for the next layer containing more relevant information required for the motion quality prediction.

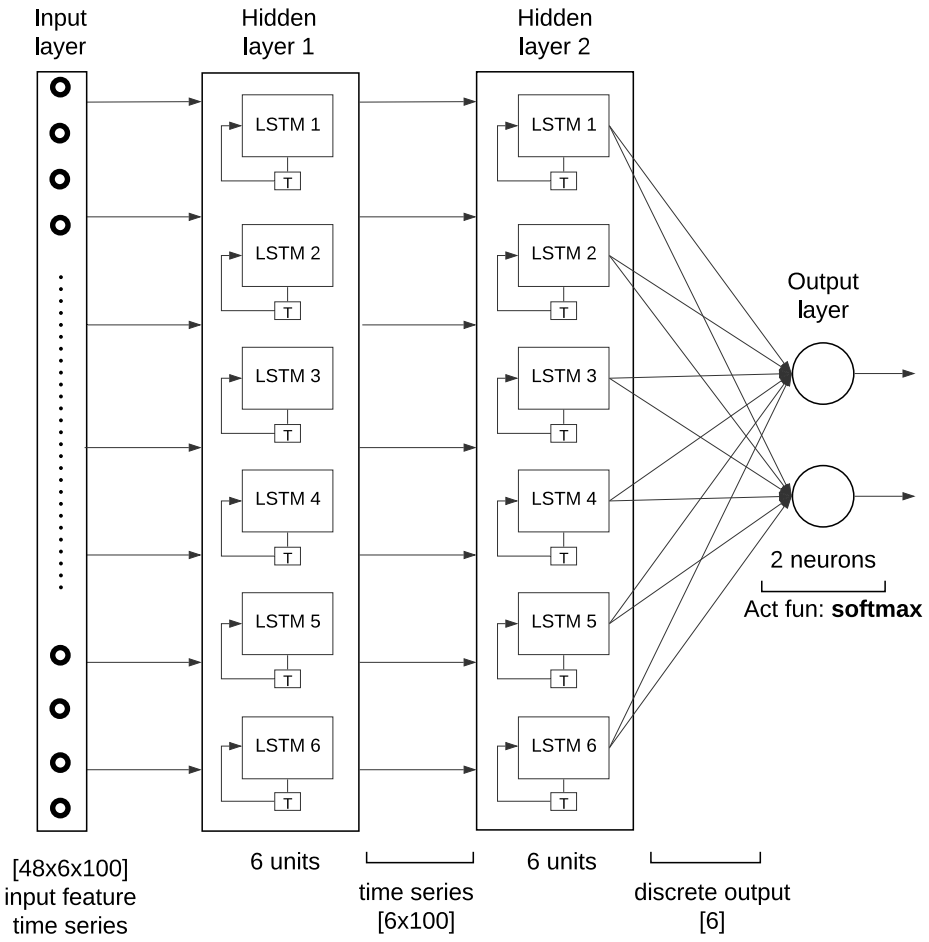


Figure 5.26: LSTM, iteration 3, accuracy 68.75%

The accuracy produced by the network is 68.75% by misclassifying 15 patients. We expected an increase in accuracy, but the accuracy decreases as compared to the previous, more simpler, network. By analyzing the bias-variance tradeoff of the network with the prediction results, it can be inferred that the bias of the network has increased more as compared to the variance. Therefore, in the next iterations, we can aim to add more layers to increase the variance and carefully change the number of LSTM units to have the right balance between the bias and variance of the network.

After numerous trials with different networks, the network in Figure 5.27 produced the highest accuracy of 77.08% by training for 100 epochs that misclassify 11 patients. The loss curve for 48 cross-validation iterations is shown in Figure 5.28, which represents a high loss on an average.

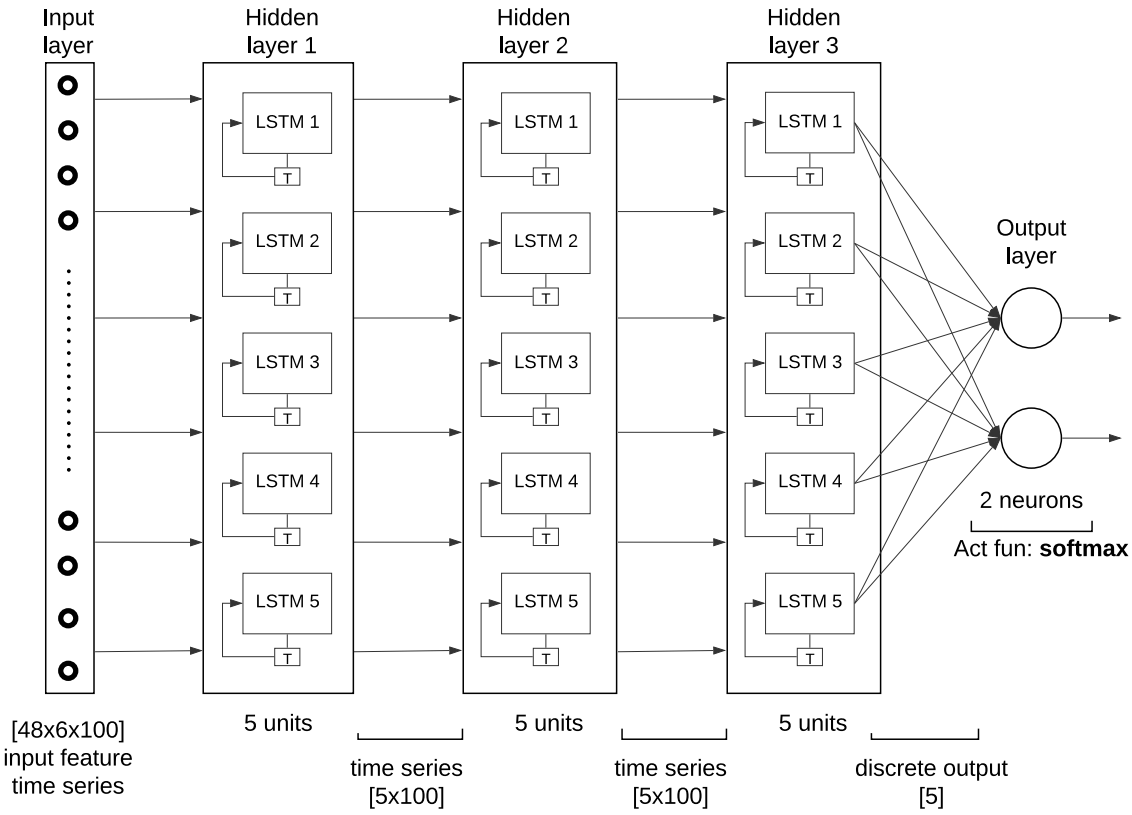


Figure 5.27: LSTM, iteration 4, accuracy 77.08%

This network contains three hidden layers with 5 LSTM units each, out of which the first two hidden layers calculates a new time series for the next subsequent layers and the third hidden layer calculates 5 discrete values for the final prediction of the motion quality.

From this analysis, we understand that LSTM networks has produced a considerably good accuracy but is low as compared to the other variants of the ANN. We cannot undermine the power of LSTM networks, but probably, LSTM require more number of data points than 48 for better prediction.

5.3.6 Merged CNN LSTM

By performing multi-class prediction using CNN, we achieve the highest accuracy of 79.16%. This demonstrates that the time-series dataset can be remodeled into a stack of heatmaps for the motion quality prediction. The undercomplete autoencoder has produced the highest accuracy of 83.33% so far, and we consider this to be the

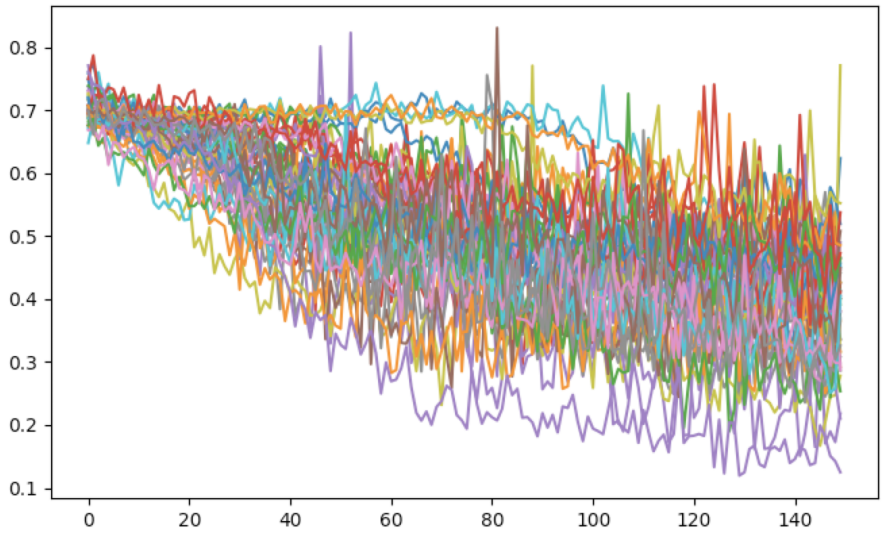


Figure 5.28: Loss curve of the 48 cross-validation training steps using the LSTM network in the final iteration

desired result as it is better than the clustering formed in the result of PCA. To achieve the accuracy with the help of CNN and LSTM networks, we try to combine the computation of CNN and LSTM as both the networks would be able to analyze the dataset by extracting different kinds of information from the heatmaps and time-series dataset respectively. Combining the information from the CNN and LSTM networks could help collect all the relevant information required for predicting the motion quality of the patients.

Figure 5.29 shows a merged CNN and LSTM network which is created after many trials with other possible networks, and has produced an accuracy of 83.33% by training for 100 epochs. For the training of this network, the dataset containing 6 inverse kinematic features is remodeled into heatmaps and 3d vectors representing images and time-series for each feature, which is then provided to the CNN and LSTM branches respectively. The CNN network is similar to the network in Figure 5.22 but has 20 CNN units that increases the CNN branch's complexity. The LSTM network is the same, as shown in Figure 5.27. Using backpropagation for learning, both the branches are tuned simultaneously where the CNN analyses the important image fragments that could be useful for the prediction and LSTM analyses the time-series information useful of the prediction. As both the branches learns together,

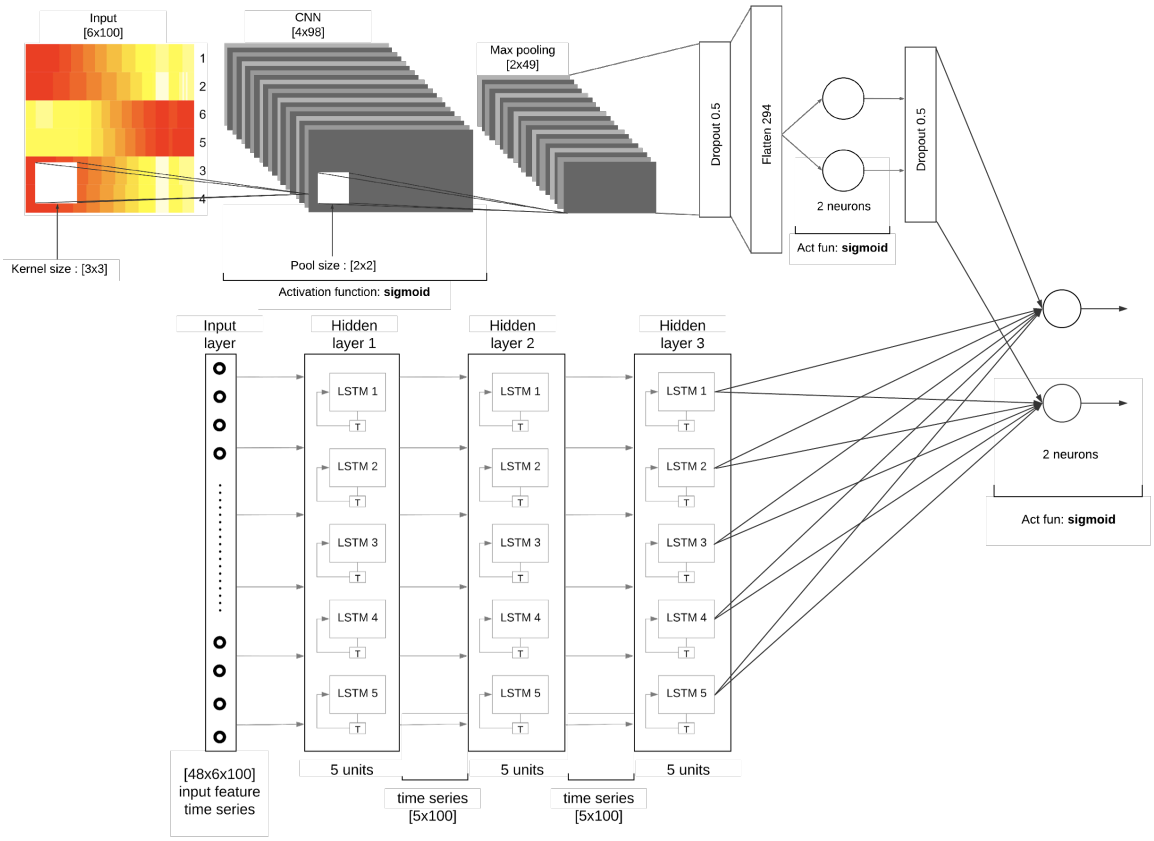


Figure 5.29: Merged CNN LSTM, accuracy 83.33%

each branch separately produces unique results that the other branch may not be able to process and combining the information from both the branches creates a pool of relevant information that helps predict the motion quality.

Figure 5.30 represents the loss curve for all 48 cross-validation iterations performed for training the network. From the plot, we understand that on an average, the loss is decreasing for all 48 iterations, but still has a higher magnitude. The network produces an accuracy of 83.33% with such a training loss, which suggests about the high capability of the network. The network, possibly, can perform with higher accuracy and lower loss if trained with a bigger dataset than used for the research.

This analysis has demonstrated a successful application of merged CNN and LSTM network that can classify the patients better than the clustering produced by the component analysis techniques. From the merged network, different aspects of the

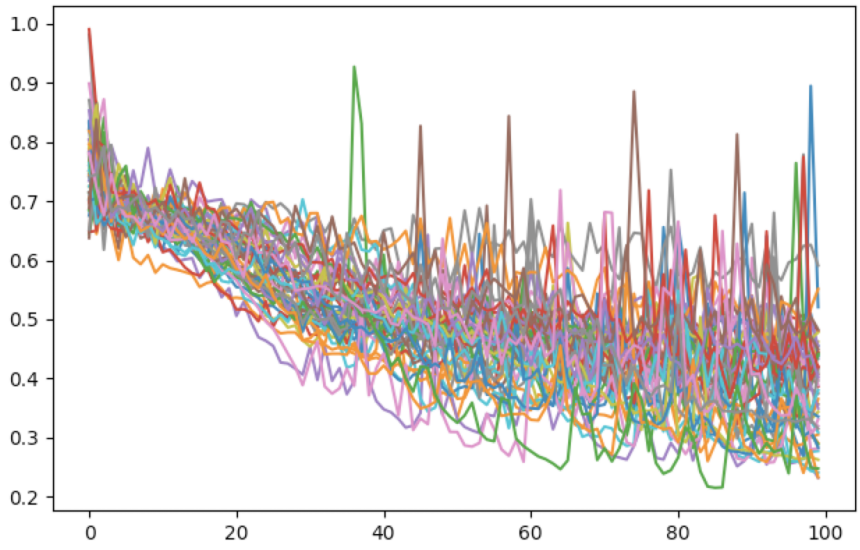


Figure 5.30: Loss curve of the 48 cross-validation training steps using the merged CNN LSTM network

original dataset can be recorded, which can be helpful for multi-class classification problems.

Chapter 6

Conclusions and Future Work

The process of effective motion quality quantification and classification is explored in the research. The results of which were discovered in the inverse kinematics data. Different variants of ANN like autoencoders, feedforward networks, CNN and LSTM have shown efficient learning using the wide shaped data by predicting the motion quality with the highest accuracy of 83.33%. ANN not only performs better than the conventional regression analysis technique, logistic regression but also produces better results than component analysis. This research also demonstrates how a time-series can be molded into different forms and utilized by multiple ML algorithms. Overall, this research has used new techniques to work on wide datasets having a variance for effective analysis and neural network training that will be useful for the researchers working in data science and machine learning field of study.

Machine learning and data science algorithms require a bigger dataset than the one used for this research. A smaller dataset restricts us to use validation dataset for neural network training, and for the dataset used, a single misclassification costs us a loss of 2.08% accuracy. For improving the results, the simulation of synthetic data should be done by calculating the deterministic time series musculoskeletal features. This will also help in predicting continuous valued inferences or by increasing the number of classes of the musculoskeletal inferences in an application.

This research has demonstrated the classification and prediction of motion quality of THA patients for one "sit-to-stand" motion cycle but is not restricted for only THA patients and motion quality. Musculoskeletal data of any group of people can

be utilized to analyze more musculoskeletal inferences like motion quality by using the data science and ANN approaches used in this research work.

Finding an effective neural network can be difficult and time-consuming. Therefore automated scripting can be helpful for building and training on all the possible neural networks.

Bibliography

[1] K. Fingar, C. Stocks, A. Weiss, and C. Steiner. "Most Frequent Operating Room Procedures Performed in U.S. Hospitals, 2003-2012." *HCUP Statistical Brief #186*, Rockville, MD: Agency for Healthcare Research and Quality, December 2014.

[2] M. P. Kadaba, H. K. Ramakrishnan, and M. E. Wootten. "Measurement of lower extremity kinematics during level walking". *Journal of Orthopaedic Research*, 8 (3): 383–392, May 1990.

[3] C. A. Myers, P. Laz, K. Shelburne, D. Judd, D. Huff, J. Winters, J. Stevens-Lapsley, P. Rullkoetter. "The impact of hip implant alignment on muscle and joint loading during dynamic activities." *Clinical Biomechanics 53* , 93–100, 2018.

[4] A. Patel, R. Wagle, M. Usrey, M. Thompson, S. Incavo, P. Noble. "Guidelines for implant placement to minimize impingement during activities of daily living after total hip arthroplasty". *The Journal of Arthoplasty*, Volume 25, Issue 8, Pages 1275–1281.e1, December 2010.

[5] W. McCulloch, W. Pitts. "A Logical Calculus of Ideas Immanent in Nervous Activity". *Bulletin of Mathematical Biophysics*, 5 (4): 115–133, 1943.

[6] S. Russell, P. Norvig. "Artificial Intelligence: A Modern Approach." *Prentice Hall*, Third edition, 2010.

[7] M. Minsky, S. Papert. "Perceptrons: An Introduction to Computational Geometry." *MIT Press*, 1969.

[8] F. Rosenblatt. "The Perceptron: A Probabilistic Model For Information Storage And Organization In The Brain". *Psychological Review*, 65 (6): 386–408, 1958.

[9] T. Mitchell. "The need for biases in learning generalizations." 1980.

[10] P. Werbos. "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences." *Harvard University*, 1975.

[11] National Center for Medical Rehabilitation Research. URL: http://opensim.stanford.edu/. Stanford University.

[12] Bertec. URL: https://www.bertec.com/products/force-plates. Columbus, OH, USA.

[13] Vicon. URL: https://www.vicon.com/. Centennial, CO, USA.

[14] CSMI. URL: http://www.csmisolutions.com/products/isokinetic-extremity-systems/humac-norm. Stoughton, MA, USA.

[15] Biodex Medical Systems. URL: https://www.biodex.com/. Shirley, NY, USA.

[16] J. Evans, P. Evans, W. Walker, A. Blom, R. Whitehouse, A. Sayers. "How long does a hip replacement last? A systematic review and meta-analysis of case series and national registry reports with more than 15 years of follow-up." *The Lancet*, 393 (10172): 647–654, 2019.

[17] M. Heller, G. Bergmann, G. Deuretzbacher, L. Claes, N. Haas, G. Duda. "Influence of femoral anteversion on proximal femoral loading: measurement and simulation in four patients." *Clin Biomech; Bristol, Avon*, 16(8):644-9, 2001.

[18] A. Waligora, N. Johanson, B. Hirsch. "Clinical Anatomy of the Quadriceps Femoris and Extensor Apparatus of the Knee." *Clinical Orthopaedics and Related Research*, 467(12): 3297–3306, 2009.

[19] P. Merriaux, Y. Dupuis, R. Boutteau, P. Vasseur, X. Savatier. "A Study of Vicon System Positioning Performance." *Sensors*, 17, 1591, 2017.

[20] Hip Joint Anatomy. URL: https://reference.medscape.com/. WebMD, New York, USA.

[21] S. Hochreiter, J. Schmidhuber. "Long Short-Term Memory." *Neural Computation*, 9(8):1735-1780, 1997.

[22] A. Hill. "The heat of shortening and dynamics constants of muscles". *Proc. R. Soc. Lond. B. London: Royal Society*, 126 (843): 136–195, 1938.

[23] R. Harshman. "Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis." *UCLA Working Papers in Phonetics*, 16: 84. No. 10,085, 1970.

[24] Y. LeCun. "Gradient-Based Learning Applied to Document Recognition." *PROC. OF THE IEEE*, 1988.

[25] A. Krizhevsky, I. Sutskever, G. E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks." *COMMUNICATIONS OF THE ACM*, Vol. 60, No. 6, 2017.

[26] C. Boyd, M. Tolson, W, Copes. "Evaluating trauma care: The TRISS method. Trauma Score and the Injury Severity Score." *The Journal of Trauma*, 27 (4): 370–378, 1987.

[27] P. Verhulst. "Notice sur la loi que la population poursuit dans son accroissement." *Correspondance mathématique et physique*, 10: 113–121, 1838 (Retrieved 3 December 2014).

[28] D. Wetcher-Hendricks. "Analyzing Quantitative Data: An Introduction for Social Researchers", p.288.

[29] R. Fisher. "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics*, 7 (2): 179–188, 1936.

[30] S. Roweis, L. Saul. "Nonlinear Dimensionality Reduction by Locally Linear Embedding." *Science*, 290 (5500): 2323–2326, 2000.

[31] K. Pearson. "On Lines and Planes of Closest Fit to Systems of Points in Space." *Philosophical Magazine*, 2 (11): 559–572, 1901.

[32] H. Hotelling. "Analysis of a complex of statistical variables into principal components." *Journal of Educational Psychology*, 24, 417–441, and 498–520, 1933.

[33] H. Hotelling. "Relations between two sets of variates." *Biometrika*, 28 (3/4): 321–377, 1936.

[34] R. Harshman. "Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis." *UCLA Working Papers in Phonetics*, 16: 84. No. 10,085, 1970.

[35] G. Cybenko. "Approximation by Superpositions of a Sigmoidal function." *In van Schuppen, Jan H. Mathematics of Control, Signals, and Systems*, Springer International, pp. 303–314, 2006.

[36] J. Snyman. "Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms." *Springer Science & Business Media*, 2005.

[37] S. Amari. "A Method of Statistical Neurodynamics." *Kybernetik*, 14, 201-215, 1974.

[38] S. Ibrić, J. Djuris, J. Parojčić, D. Zorica. "Artificial Neural Networks in Evaluation and Optimization of Modified Release Solid Dosage Forms." *Pharmaceutics*, 4(4):531-550, 2012.

[39] L. Bottou, F. Curtis, J. Nocedal. "Optimization Methods for Large-Scale Machine Learning." *Siam Reviews*, 60(2):223-311, 2018.

[40] C. Myers, P. Laz, K. Shelburne, B. Davidson. "A Probabilistic Approach to Quantify the Impact of Uncertainty Propagation in Musculoskeletal Simulations." *Annals of Biomedical Engineering*, Vol. 43, No. 5, May 2015.

[41] M. Cilla, E. Borgiani, J. Martínez, G. Duda, S. Checa. "Machine learning techniques for the optimization of joint replacements: Application to a short-stem hip implant." *PLOS ONE*, 12(9): e0183755, 2017.

[42] P. Danaee, R. Ghaeini, D. Hendrix. "A Deep Learning Approach for Cancer Detection and Relevant Gene Identification." *Pacific Symposium on Biocomputing*, 2017.

[43] C. Floyd, J. Lo, A. Yun, D. Sullivan, P. Kornguth. "Prediction of breast cancer malignancy using an artificial neural network." *Cancer*, 74(11):2944-8, 1994.

[44] A. Phinyomark, G. Petri, E. Ibáñez-Marcelo, S. Osis, R. Ferber. "Analysis of Big Data in Gait Biomechanics: Current Trends and Future Directions." *Journal of Medical and Biological Engineering*, Volume 38, Issue 2, pp 244–260, 2018.

[45] R. Kirkwood, R. Resende, C. Magalhães, H. Gomes, S. Mingoti, R. Sampaio. "Application of principal component analysis on gait kinematics in elderly women with knee osteoarthritis." *Rev Bras Fisioter*, 15(1):52-8, 2011.

[46] N. Helwig, S. Hong, J. Polk, M. Lague. "Analysis of gait cycle shapes using Parallel Factor Analysis." *Proceedings of the 34th Annual Meeting of the American Society of Biomechanics*, p. 84, 2010.

[47] E. Hsiao-Wecksler, J. Polk, K. Rosengren, J. Sosnoff, S. Hong. "A Review of New Analytic Techniques for Quantifying Symmetry in Locomotion." *Symmetry*, 2, 1135-1155, 2010.

Appendix

A.1 Building ANN - Bias Variance Dilemma

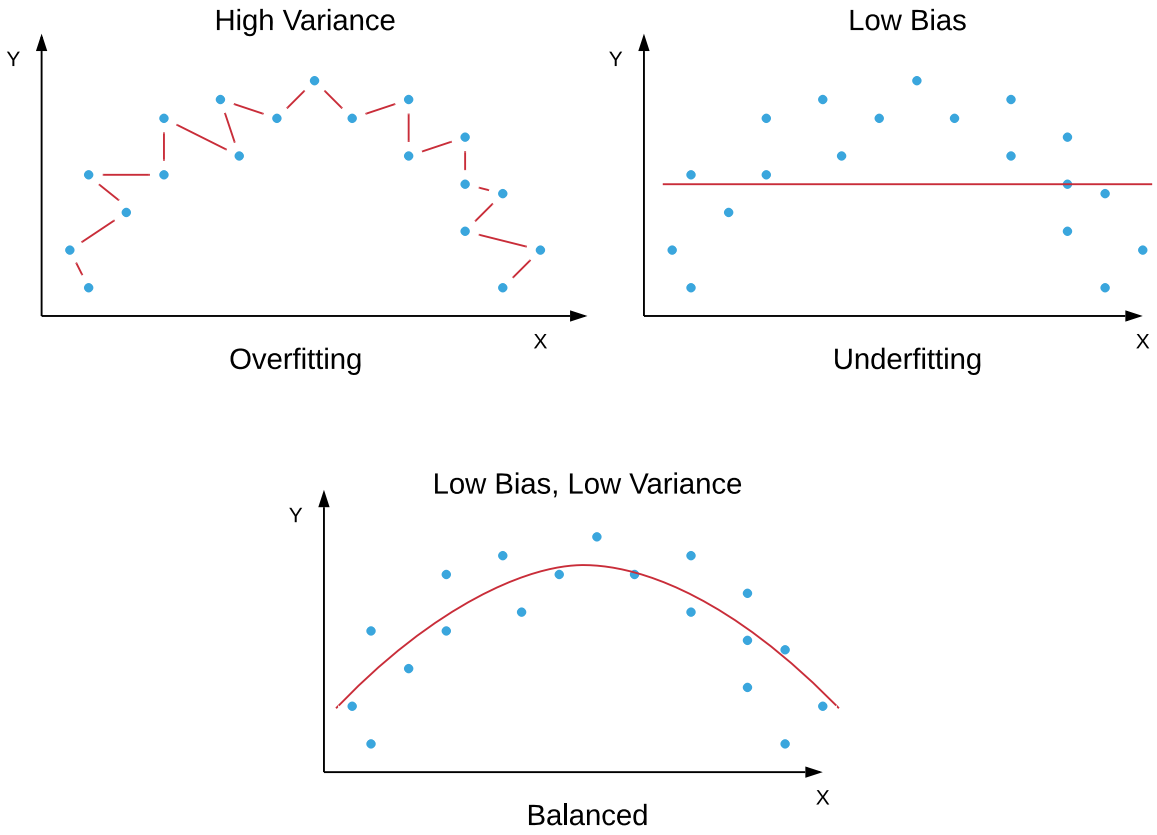


Figure 1: Bias-variance dilemma

The application of ANN in this research is based on the prediction of multi-class classified information. Therefore, the training errors are essential to understand while building a network progressively by extending the computation of the network with the help of including more neurons and layers. The learning errors can be understood with the help of bias and variance of the learning curve generated from a network. A proper understanding of bias and variance helps in eliminating issues like underfitting and overfitting.

Bias and variance are two properties of a curve that explains the complexity of an unknown function which is created by the clusters of neurons. Bias is the difference between the network's prediction and the actual value to be predicted. The high

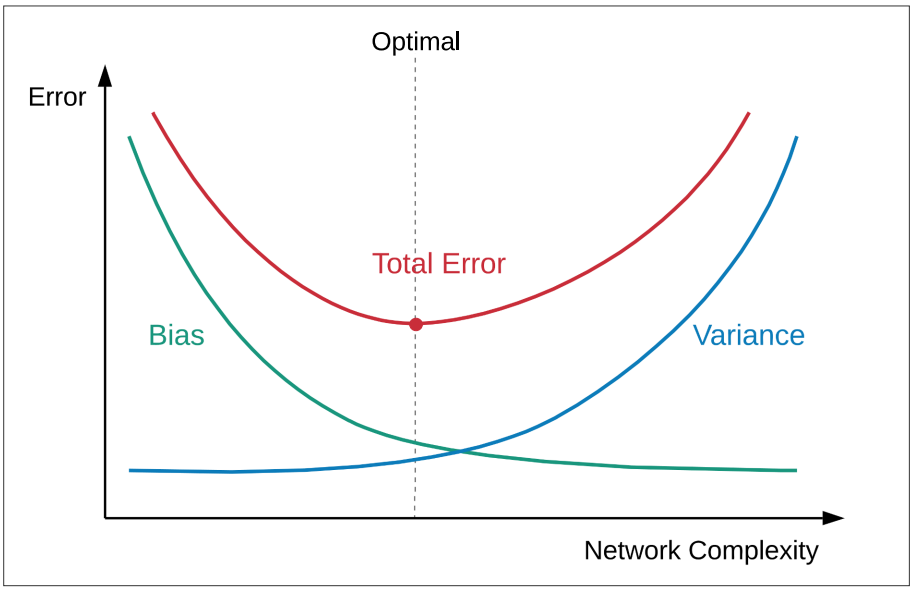


Figure 2: Bias-variance error

bias of the network pays little attention to the training dataset and leads to a high error and a low accuracy of the overall prediction. Variance is the variability of the network's unknown function that explains the spread of the function for a given training dataset. High variance pays more attention to the training dataset and cannot generalize the unseen testing dataset. Due to this, the training errors will be low, but the testing error will be high that can produce less accurate predictions. Therefore, there should be a right balance between bias and variance to construct a better ANN model.

The total error of a network is the sum of the squared bias and variance.

$$Error = Bias^2 + Variance$$

Therefore, there is a bias and variance tradeoff for a model to minimize the overall training and testing errors. As bias increases, variance decreases and vice versa. Therefore, a better model is the one that has a balanced bias and variance.

While building a neural network, bias and variance dilemma is taken into consideration for extending the complexity of a network. To verify the total error during

training a network, we check the loss during each epoch and stop the training until minimum error is reached.

A.2 Medical Terms

A.2.1 Osteoarthritis

Osteoarthritis is a type of arthritis which occurs due to wearing down of the flexible tissues at the end of the bones.

A.2.2 Inflammatory arthritis

Inflammatory arthritis occurs due to inflammation of multiple joints resulting into pain and stiffness.

A.2.3 Osteonecrosis

Osteonecrosis, also known as avascular necrosis, is a bone disease that results in the bone tissue dying because of a lack of blood supply.

A.2.4 Patellofemoral joint

The patellofemoral joint is the meeting point of the patella(knee cap) and femur(thigh) at the knee front in the human body.

A.3 Variable Description

A.3.1 Inverse kinematics

1. **Pelvis tilt**
    - Unit: degrees
    - The angle of how much pelvis is tilted in sagittal plane w.r.t anteroposterior axis
    - Pelvis tilt is the angle by which the pelvis is bent in front or back of human body orientation
2. **Pelvis list**
    - Unit: degrees

- The angle of how much pelvis is tilted in frontal plane w.r.t mediolateral axis
- Quantified using the angle between the horizontal and a line connecting the anterior superior iliac spine and the posterior superior iliac spine.

3. **Pelvis rotation**
   - Unit: degrees
   - The angle of pelvis swing on horizontal plain w.r.t mediolateral axis
   - A pelvic rotation is when a one anterior superior iliac spine is more forward than the other.
4. **Pelvis Tx**
   - Unit: meters
   - Pelvis translation coordinate along x axis
5. **Pelvis Ty**
   - Unit: meters
   - Pelvis translation coordinate along y axis
6. **Pelvis Tz**
   - Unit: meter
   - Pelvis translation coordinate along z axis
7. **Hip flexion r**
   - Unit: degrees
   - Angle of bending in the right hip joint
8. **Hip flexion l**
   - Unit: degrees
   - Angle of bending in the left hip joint
9. **Hip adduction r**
   - Unit: degrees
   - Angle during bending right hip joint towards its normal position
10. **Hip adduction l**
    - Unit: degrees
    - Angle during bending left hip joint towards its normal position

11. **Hip rotation r**
    - Unit: degrees
    - Angle of rotation of right hip joint in horizontal plain
12. **Hip rotation l**
    - Unit: degrees
    - Angle of rotation of left hip joint in horizontal plain
13. **Knee flexion r**
    - Unit: degrees
    - Angle of bending in the right knee joint
14. **Knee flexion l**
    - Unit: degrees
    - Angle of bending in the left knee joint
15. **Ankle angle r**
    - Unit: degrees
    - Angle of bending in the right ankle joint
16. **Ankle angle l**
    - Unit: degrees
    - Angle of bending in the left ankle joint
17. **Lumbar extension**
    - Unit: meters
    - Length of spine extension during spine bending
18. **Lumbar bending**
    - Unit: degrees
    - Angle of bending of spine in the frontal plain
19. **Lumbar rotation**
    - Unit: degrees
    - Angle of the swing of the spine along the vertical axis
20. **Ground reaction force x r**
    - Unit: newtons

- Force exerted between right feet and ground along the x axis

21. **Ground reaction force y r**
    - Unit: newtons
    - Force exerted between right feet and ground along the y axis
22. **Ground reaction force z r**
    - Unit: newtons
    - Force exerted between right feet and ground along the z axis
23. **Ground reaction force x l**
    - Unit: newtons
    - Force exerted between left feet and ground along the x axis
24. **Ground reaction force y l**
    - Unit: newtons
    - Force exerted between left feet and ground along the y axis
25. **Ground reaction force z l**
    - Unit: newtons
    - Force exerted between left feet and ground along the z axis
26. **Center of pressure x r**
    - Unit: meters
    - Coordinate on x axis for center of pressure on pressure plates underneath right feet
27. **Center of pressure y r**
    - Unit: meters
    - Coordinate on y axis for center of pressure on pressure plates underneath right feet
28. **Center of pressure x l**
    - Unit: meters
    - Coordinate on x axis for center of pressure on pressure plates underneath left feet
29. **Center of pressure y l**
    - Unit: meters

- Coordinate on y axis for center of pressure on pressure plates underneath left feet

30. **Mom r**
    - Unit: newton meters
    - Moment or torque resulting from force along the length of right leg
31. **Mom l**
    - Unit: newton meters
    - Moment or torque resulting from force along the length of left leg

A.3.2 Inverse dynamics

1. **Hip load x r**
    - Unit: newtons
    - Load on the right hip joint on x axis
2. **Hip load y r**
    - Unit: newtons
    - Load on the right hip joint on y axis
3. **Hip load z r**
    - Unit: newtons
    - Load on the right hip joint on z axis
4. **Hip load x l**
    - Unit: newtons
    - Load on the left hip joint on x axis
5. **Hip load y l**
    - Unit: newtons
    - Load on the left hip joint on y axis
6. **Hip load z l**
    - Unit: newtons
    - Load on the left hip joint on z axis
7. **Hip magnitude r**
    - Unit: newtons

- Overall load on the right hip joint

8. **Hip magnitude l**
    - Unit: newtons
    - Overall load on the left hip joint
9. **Knee load x r**
    - Unit: newtons
    - Load on the right knee joint on x axis
10. **Knee load y r**
    - Unit: newtons
    - Load on the right knee joint on y axis
11. **Knee load z r**
    - Unit: newtons
    - Load on the right knee joint on z axis
12. **Knee load x l**
    - Unit: newtons
    - Load on the left knee joint on x axis
13. **Knee load y l**
    - Unit: newtons
    - Load on the left knee joint on y axis
14. **Knee load z l**
    - Unit: newtons
    - Load on the left knee joint on z axis
15. **Knee magnitude r**
    - Unit: newtons
    - Overall load on the right knee joint
16. **Knee magnitude l**
    - Unit: newtons
    - Overall load on the left knee joint

A.3.3 Muscle force

1. **Semimembranosus force r**
   - Unit: newtons
   - The force exerted on semimembranosus muscle of right leg during a motion
2. **Semimembranosus force l**
   - Unit: newtons
   - The force exerted on semimembranosus muscle of left leg during a motion
3. **Semitendinosus force r**
   - Unit: newtons
   - The force exerted on Semitendinosus muscle of right leg during a motion
4. **Semitendinosus force l**
   - Unit: newtons
   - The force exerted on semitendinosus muscle of left leg during a motion
5. **Biceps femoris-long head force r**
   - Unit: newtons
   - The force exerted on biceps femoris-long head muscle of right leg during a motion
6. **Biceps femoris-long head force l**
   - Unit: newtons
   - The force exerted on biceps femoris-Long Head muscle of left leg during a motion
7. **Biceps femoris-short head force r**
   - Unit: newtons
   - The force exerted on biceps femoris-short head muscle of right leg during a motion
8. **Biceps femoris-short head force l**
   - Unit: newtons
   - The force exerted on biceps femoris-short head muscle of left leg during a motion
9. **Gluteus force r**

- Unit: newtons
- The force exerted on gluteus muscles of right leg during a motion
- Gluteus is a group of gluteus maximus, gluteus medius, and gluteus minimus muscles is a group of the main exterior muscles of the hip
- The force of each muscle is counted as a separate variable

10. **Gluteus force l**
    - Unit: newtons
    - The force exerted on gluteus muscles of left leg during a motion
    - Gluteus is a group of gluteus maximus, gluteus medius, and gluteus minimus muscles is a group of the main exterior muscles of the hip
    - The force of each muscle is counted as a separate variable
11. **Rectus femoris force r**
    - Unit: newtons
    - The force exerted on rectus femoris muscle of right leg during a motion
    - Rectus femoris muscle is located in the superior, anterior, and middle compartment of the thigh
12. **Rectus femoris force l**
    - Unit: newtons
    - The force exerted on rectus femoris muscle of left leg during a motion
    - Rectus femoris muscle is located in the superior, anterior, and middle compartment of the thigh
13. **Vastus medialis force r**
    - Unit: newtons
    - The force exerted on vastus medialis muscle of right leg during a motion
    - Vastus medialis muscle is located in the front of the thigh above the knee cap
14. **Vastus medialis force l**
    - Unit: newtons
    - The force exerted on vastus medialis muscle of left leg during a motion

- Vastus medialis muscle is located in the front of the thigh above the knee cap

15. **Vastus intermedius force r**
    - Unit: newtons
    - The force exerted on Vastus intermedius muscle of right leg during a motion
    - Vastus intermedius muscle is located in the anterior region of the thigh
16. **Vastus intermedius force l**
    - Unit: newtons
    - The force exerted on vastus intermedius muscle of left leg during a motion
    - Vastus intermedius muscle is located in the anterior region of the thigh
17. **Vastus lateralis force r**
    - Unit: newtons
    - The force exerted on vastus lateralis muscle of right leg during a motion
    - Vastus lateralis muscle is located on the side of the thigh
18. **Vastus lateralis force l**
    - Unit: newtons
    - The force exerted on vastus lateralis muscle of left leg during a motion
    - Vastus lateralis muscle is located on the side of the thigh
19. **Medial gastrocnemius force r**
    - Unit: newtons
    - The force exerted on medial gastrocnemius muscle of right leg during a motion
    - Medial gastrocnemius muscle is located in the outer back part of the lower leg, runs from just above the knee to the heel
20. **Medial gastrocnemius force l**
    - Unit: newtons
    - The force exerted on medial gastrocnemius muscle of left leg during a motion
    - Medial gastrocnemius muscle is located in the outer back part of the lower leg, runs from just above the knee to the heel

21. **Lateral gastrocnemius force r**
    - Unit: newtons
    - The force exerted on lateral gastrocnemius muscle of right leg during a motion
    - Lateral gastrocnemius muscle is located in the inner back part of the lower leg, runs from just above the knee to the heel
22. **Lateral gastrocnemius force l**
    - Unit: newtons
    - The force exerted on lateral gastrocnemius muscle of left leg during a motion
    - Lateral gastrocnemius muscle is located in the inner back part of the lower leg, runs from just above the knee to the heel
23. **Soleus force r**
    - Unit: newtons
    - The force exerted on soleus muscle of right leg during a motion
    - Soleus muscle is located in inner center back part of the lower leg, runs from just above the knee to the heel
24. **Soleus force l**
    - Unit: newtons
    - The force exerted on soleus muscle of left leg during a motion
    - Soleus muscle is located on the center back part of the lower leg, runs from just above the knee to the heel

A.4 PCA on mean centered inverse kinematics for all feature sets

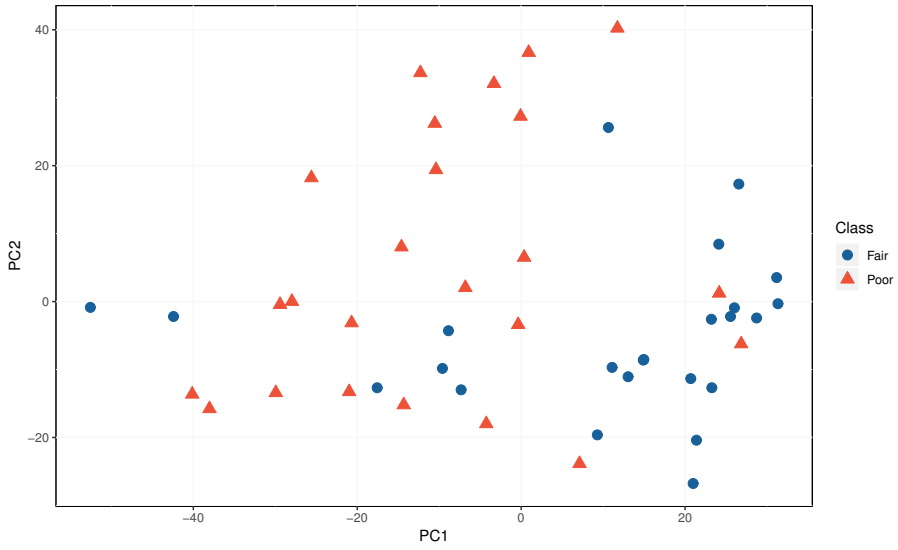


Figure 3: Clustering by PCA using inverse kinematic mean centered features and motion quality quantified using feature set 2

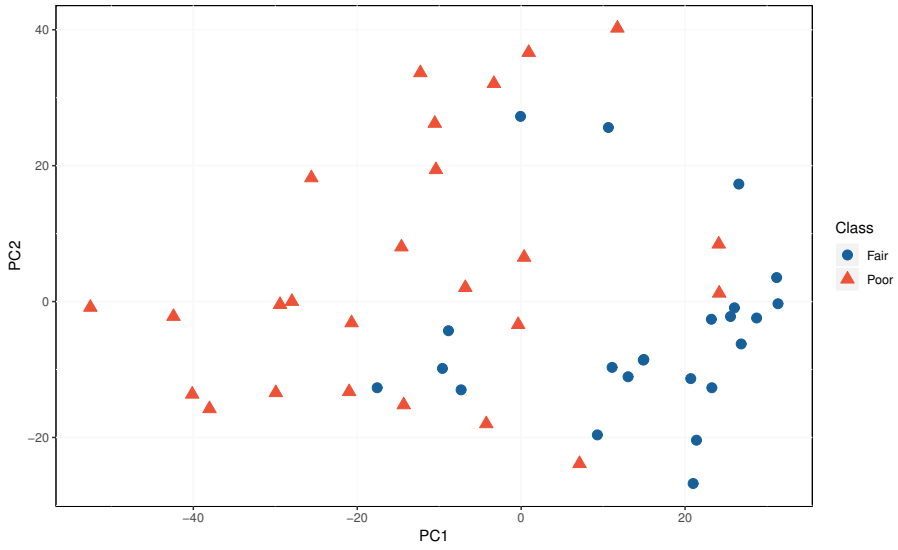


Figure 4: Clustering by PCA using inverse kinematic mean centered features and motion quality quantified using feature set 3

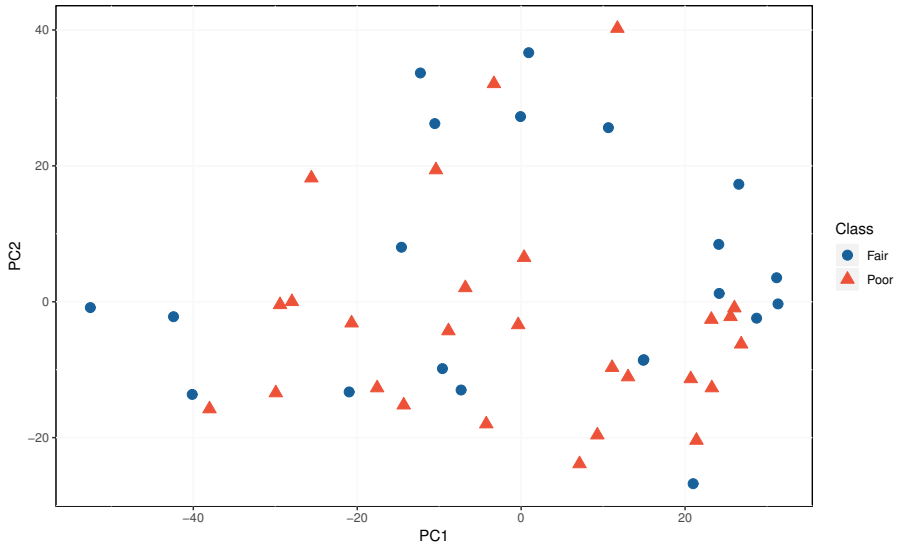


Figure 5: Clustering by PCA using inverse kinematic mean centered features and motion quality quantified using feature set 4

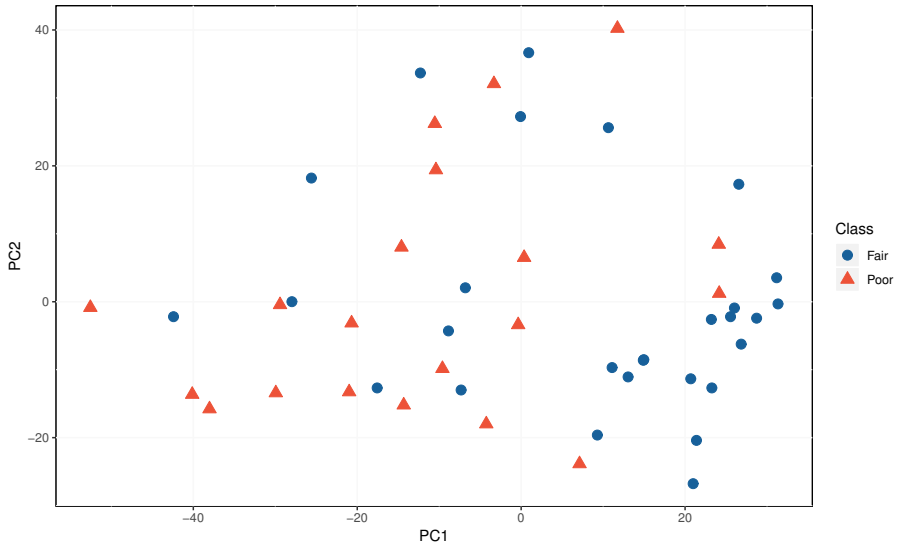


Figure 6: Clustering by PCA using inverse kinematic mean centered features and motion quality quantified using feature set 5

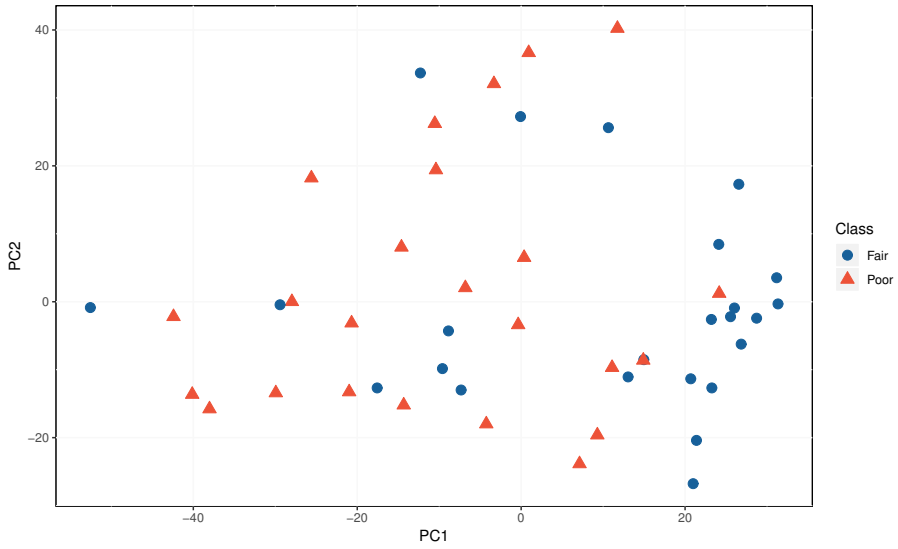


Figure 7: Clustering by PCA using inverse kinematic mean centered features and motion quality quantified using feature set 6

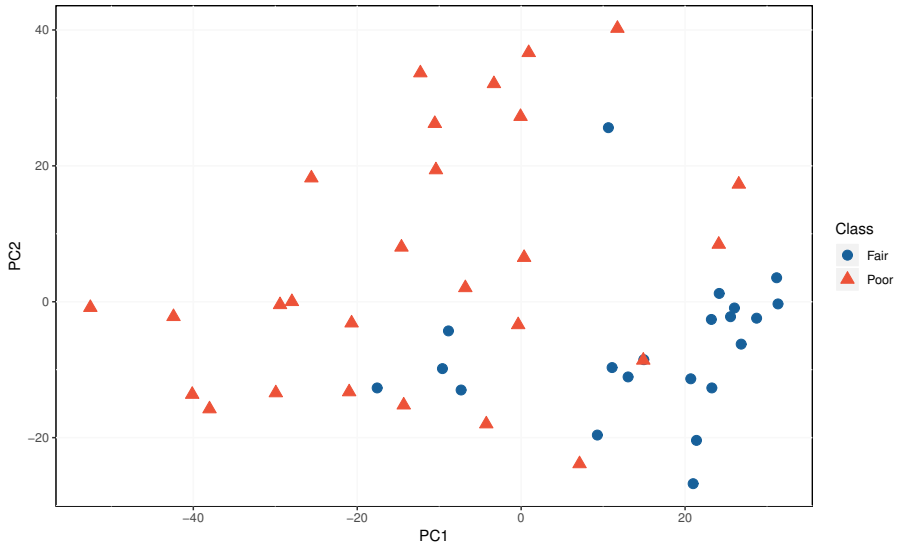


Figure 8: Clustering by PCA using inverse kinematic mean centered features and motion quality quantified using feature set 7

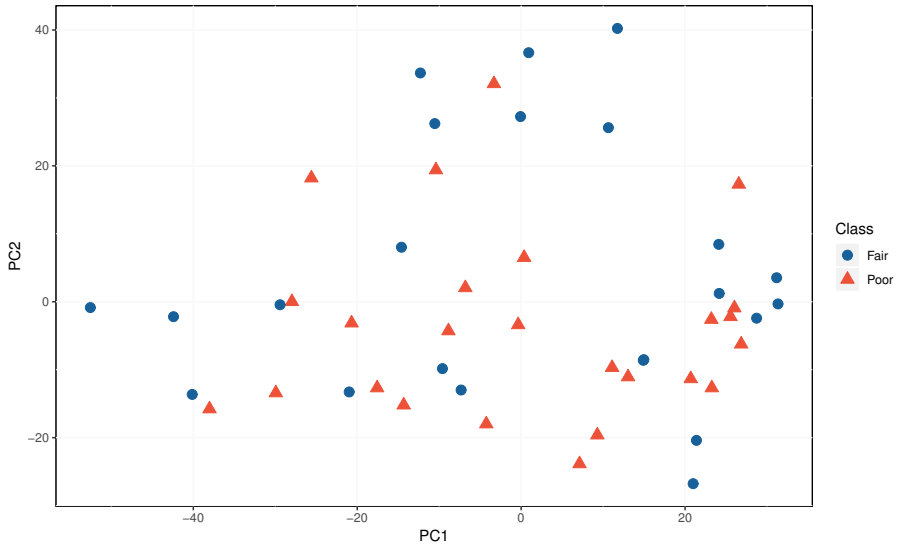


Figure 9: Clustering by PCA using inverse kinematic mean centered features and motion quality quantified using feature set 8

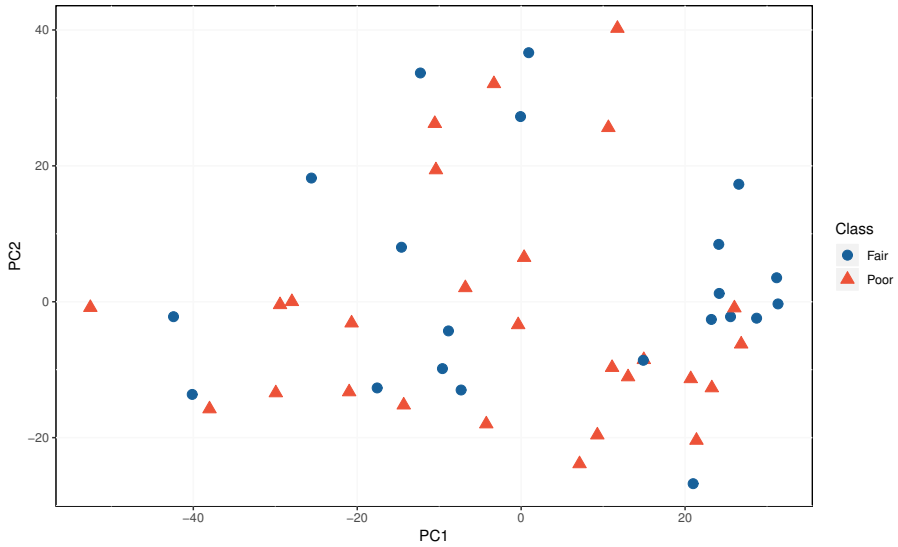


Figure 10: Clustering by PCA using inverse kinematic mean centered features and motion quality quantified using feature set 9

A.5 PCA on relative variation scaled inverse kinematics for all feature sets

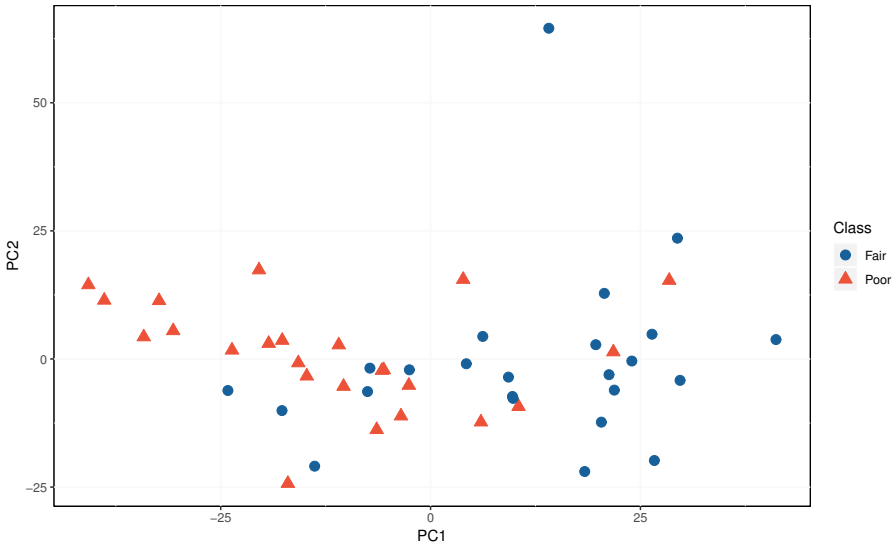


Figure 11: Clustering by PCA using inverse kinematic relative variation scaled features and motion quality quantified using feature set 2

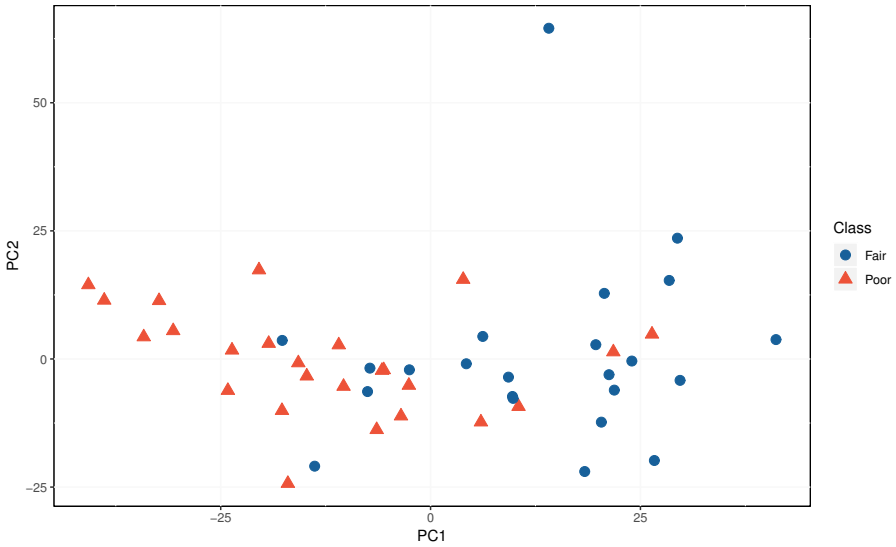


Figure 12: Clustering by PCA using inverse kinematic relative variation scaled features and motion quality quantified using feature set 3

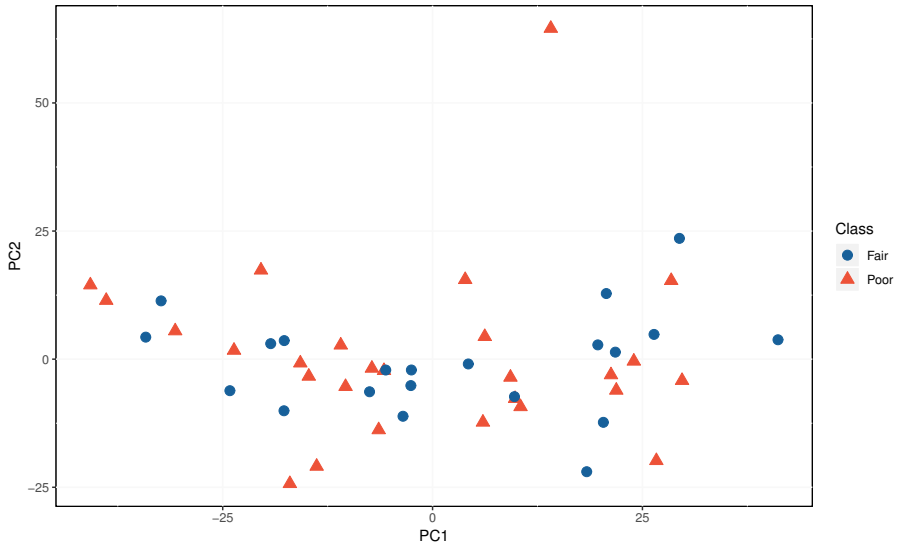


Figure 13: Clustering by PCA using inverse kinematic relative variation scaled features and motion quality quantified using feature set 4

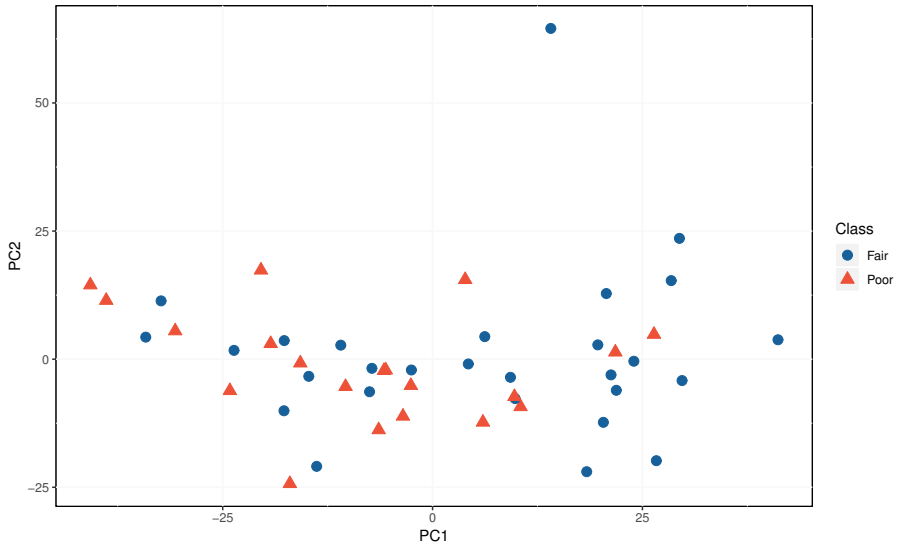


Figure 14: Clustering by PCA using inverse kinematic relative variation scaled features and motion quality quantified using feature set 5

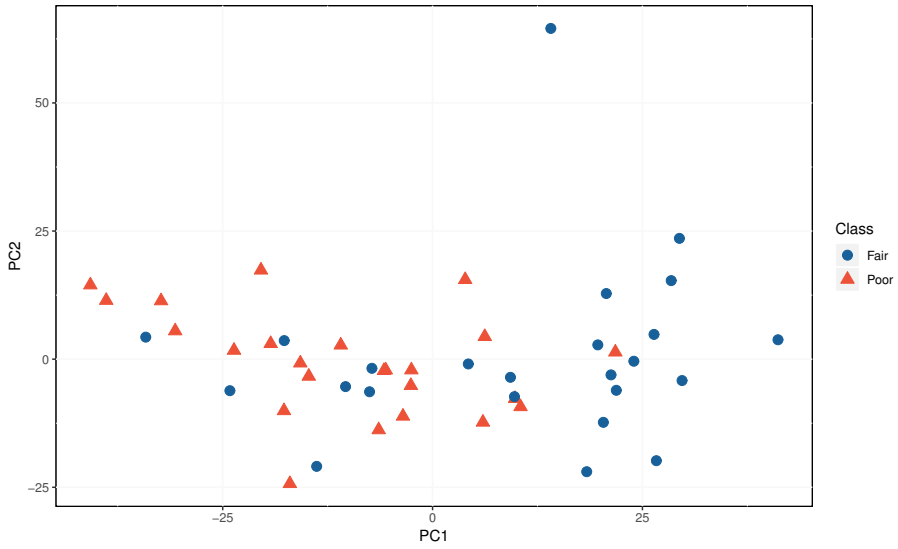


Figure 15: Clustering by PCA using inverse kinematic relative variation scaled features and motion quality quantified using feature set 6

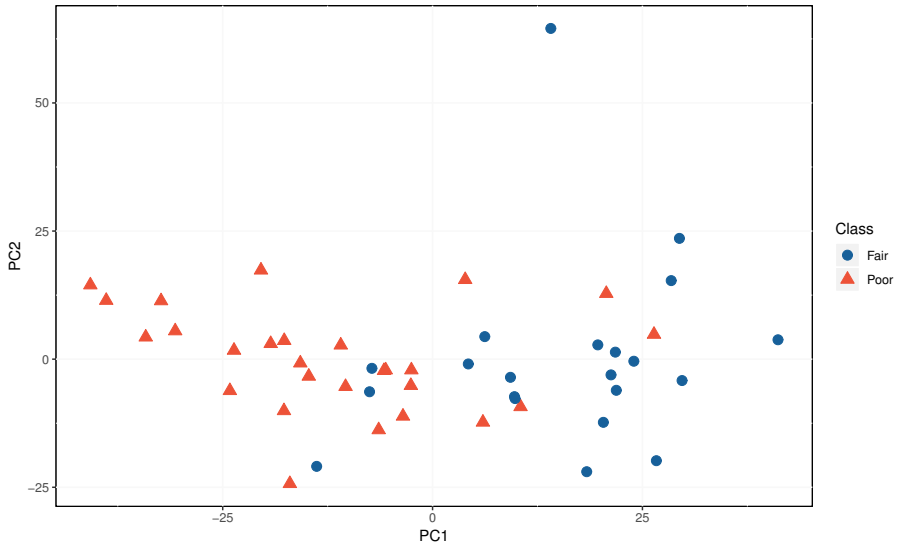


Figure 16: Clustering by PCA using inverse kinematic relative variation scaled features and motion quality quantified using feature set 7

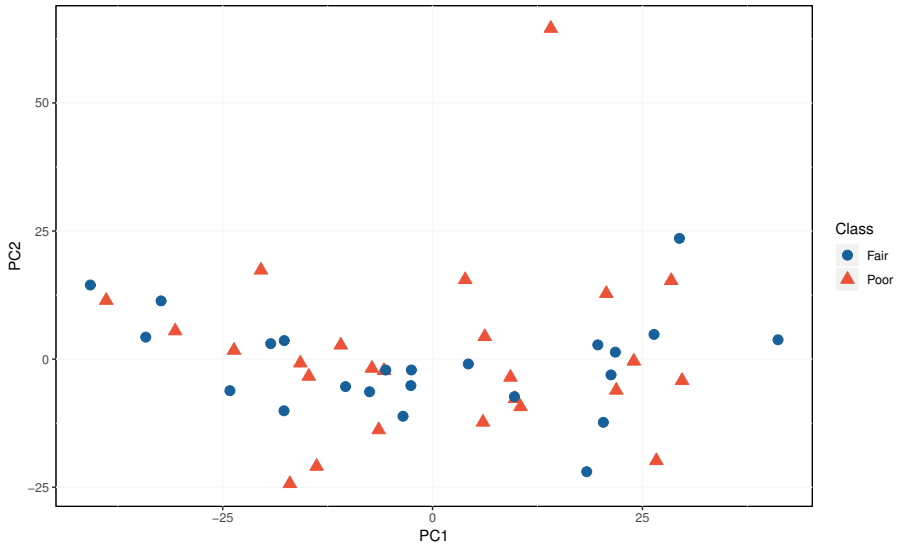


Figure 17: Clustering by PCA using inverse kinematic relative variation scaled features and motion quality quantified using feature set 8

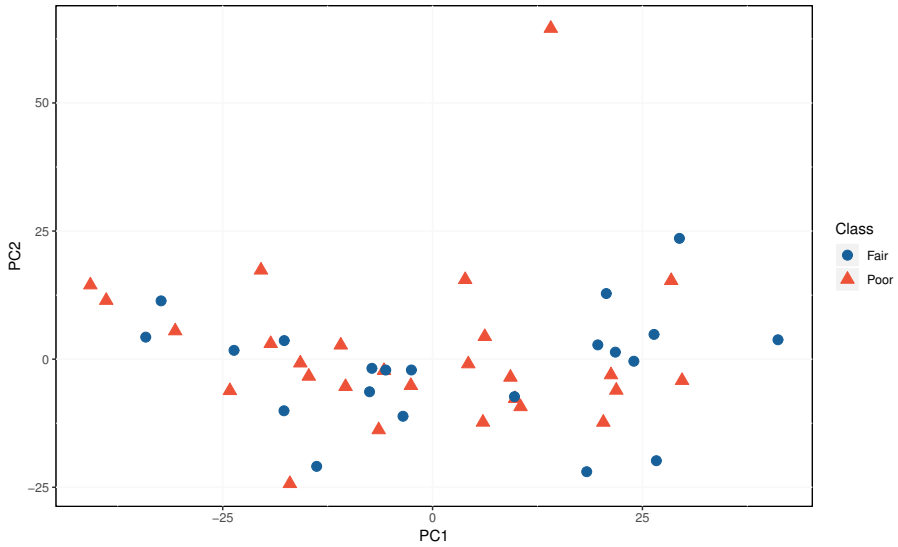


Figure 18: Clustering by PCA using inverse kinematic relative variation scaled features and motion quality quantified using feature set 9

A.6 PARAFAC on mean centered inverse kinematics for all feature sets

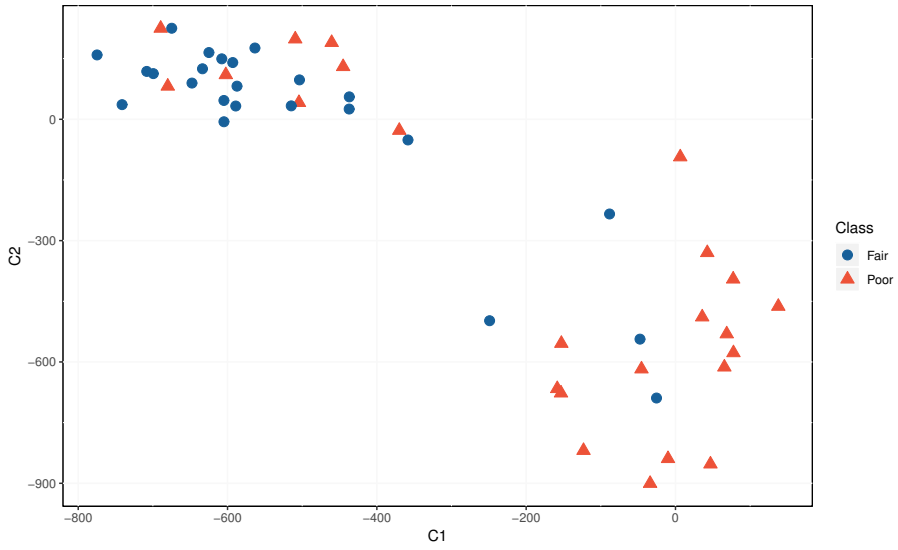


Figure 19: Clustering by PARAFAC using inverse kinematic mean centered features and motion quality quantified using feature set 2

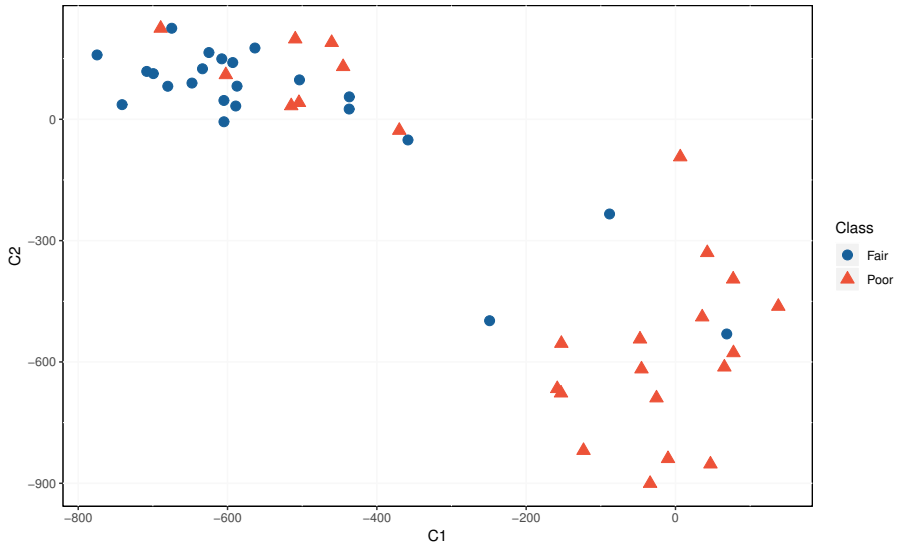


Figure 20: Clustering by PARAFAC using inverse kinematic mean centered features and motion quality quantified using feature set 3

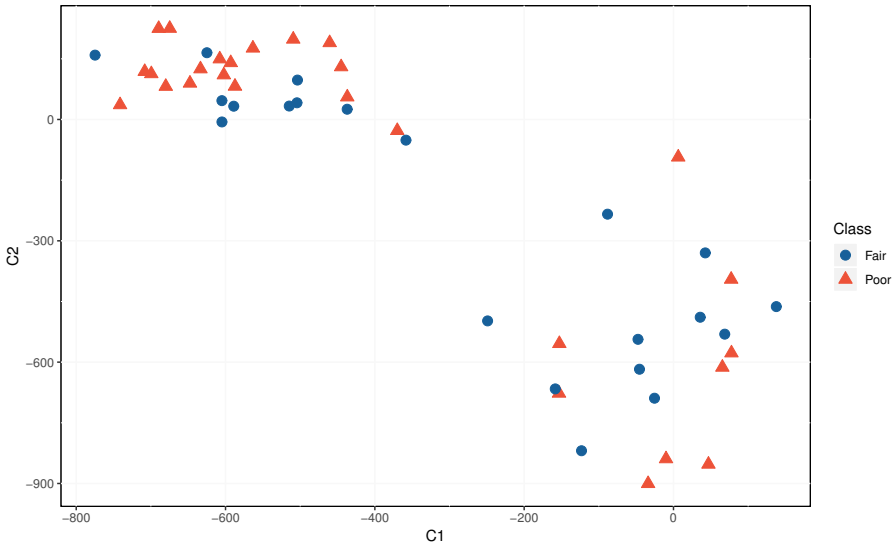


Figure 21: Clustering by PARAFAC using inverse kinematic mean centered features and motion quality quantified using feature set 4

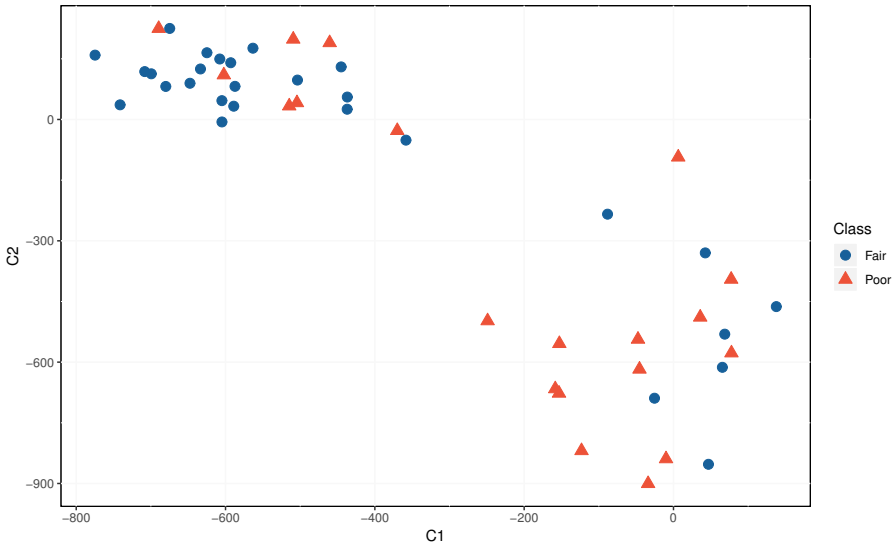


Figure 22: Clustering by PARAFAC using inverse kinematic mean centered features and motion quality quantified using feature set 5

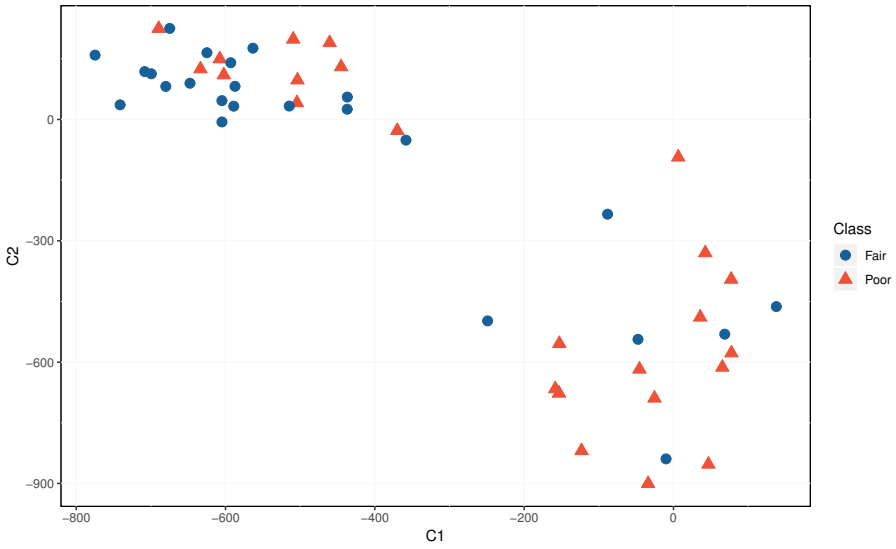


Figure 23: Clustering by PARAFAC using inverse kinematic mean centered features and motion quality quantified using feature set 6

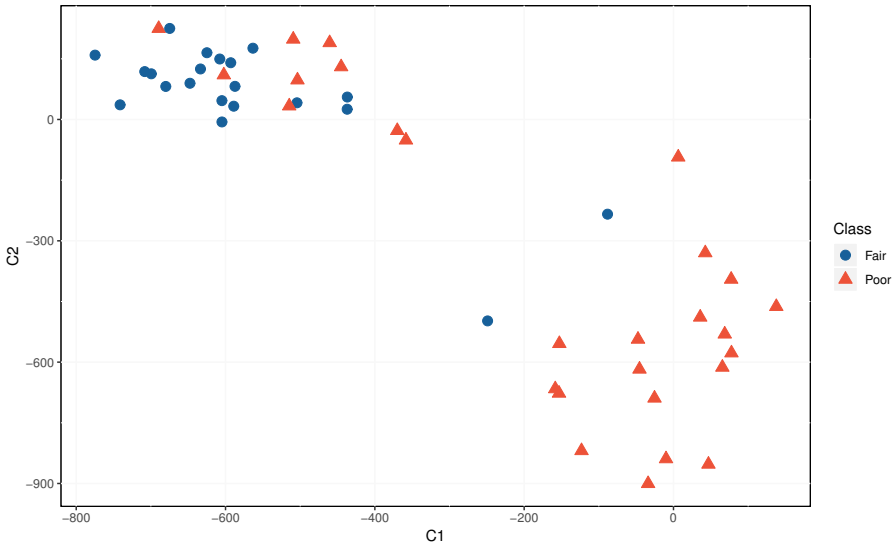


Figure 24: Clustering by PARAFAC using inverse kinematic mean centered features and motion quality quantified using feature set 7

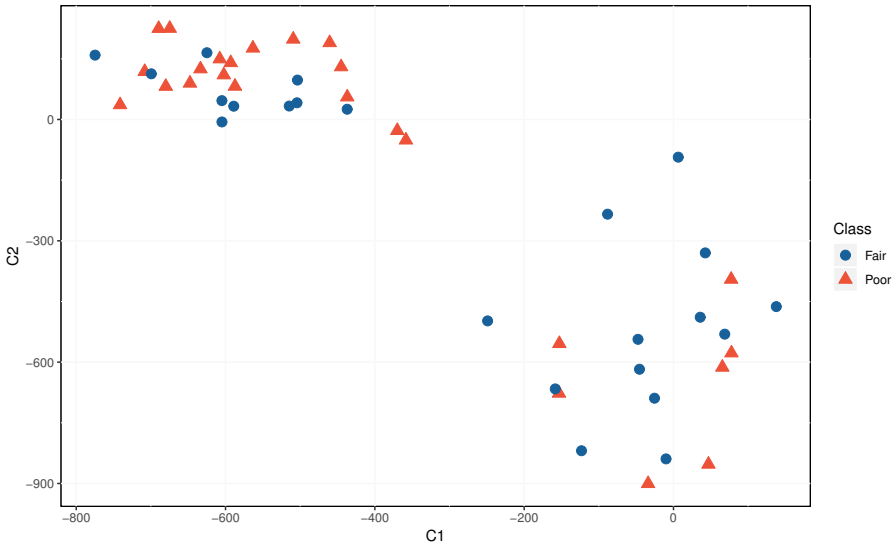


Figure 25: Clustering by PARAFAC using inverse kinematic mean centered features and motion quality quantified using feature set 8

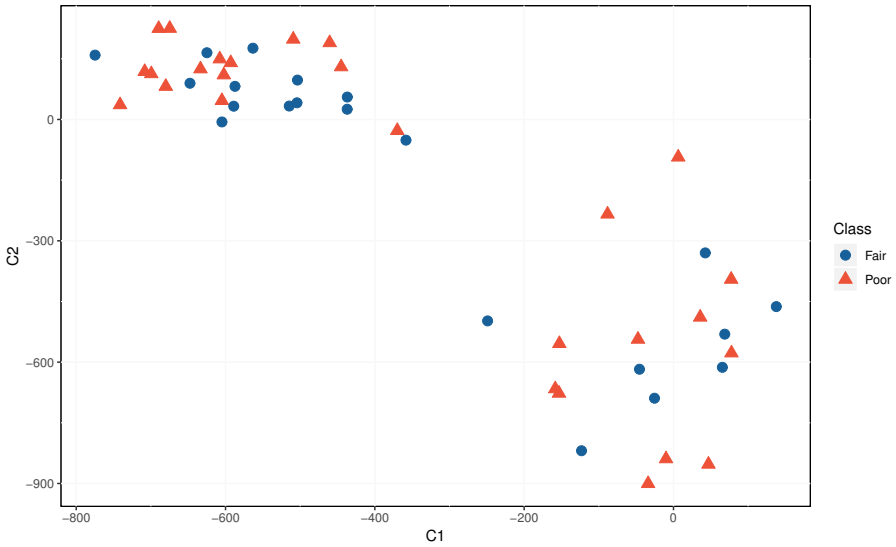


Figure 26: Clustering by PARAFAC using inverse kinematic mean centered features and motion quality quantified using feature set 9

A.7 PARAFAC on relative variation scaled inverse kinematics for all feature sets

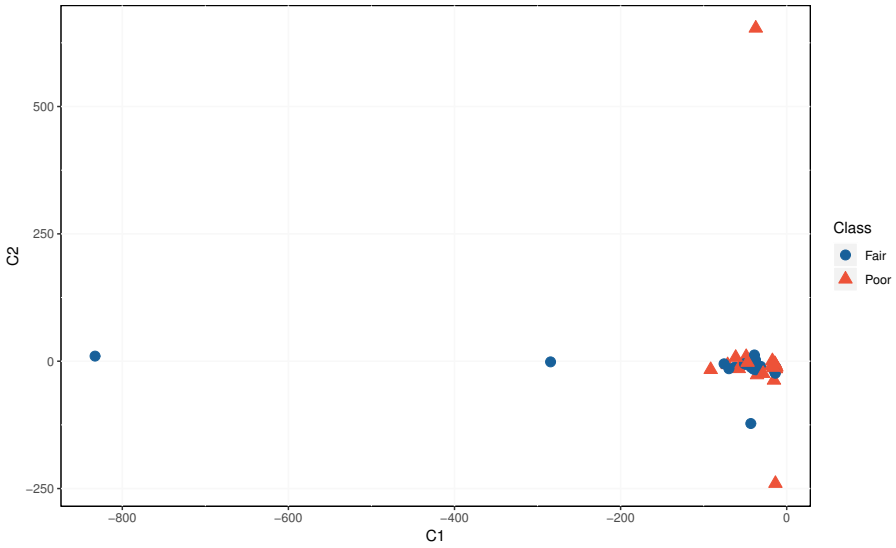


Figure 27: Clustering by PARAFAC using inverse kinematic relative variation scaled features and motion quality quantified using feature set 2

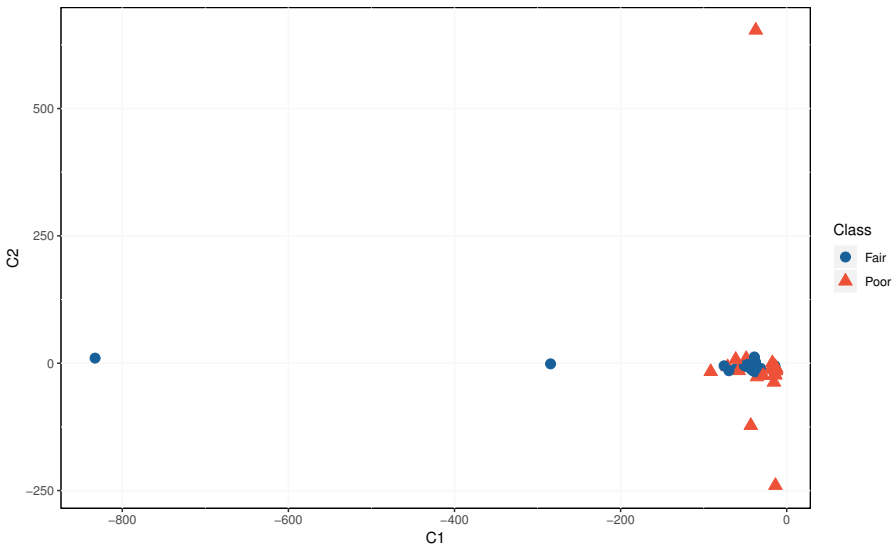


Figure 28: Clustering by PARAFAC using inverse kinematic relative variation scaled features and motion quality quantified using feature set 3

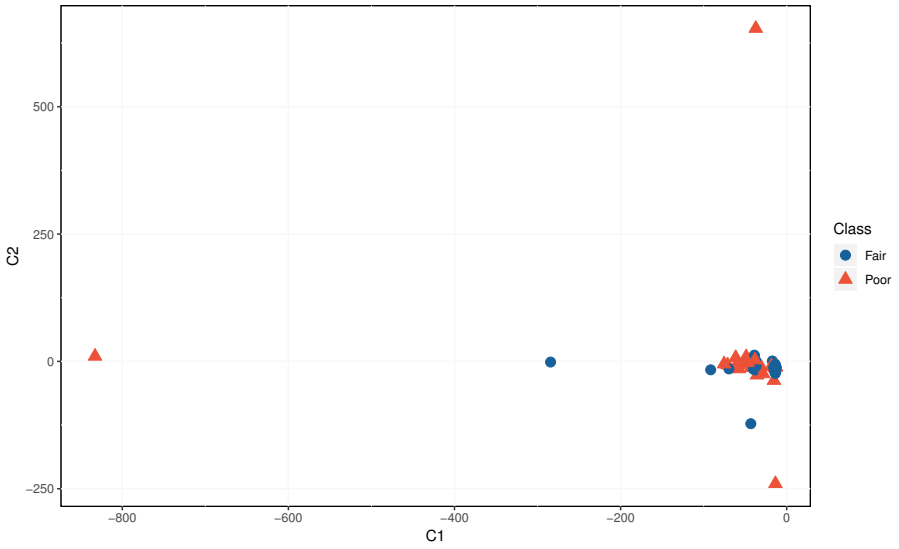


Figure 29: Clustering by PARAFAC using inverse kinematic relative variation scaled features and motion quality quantified using feature set 4

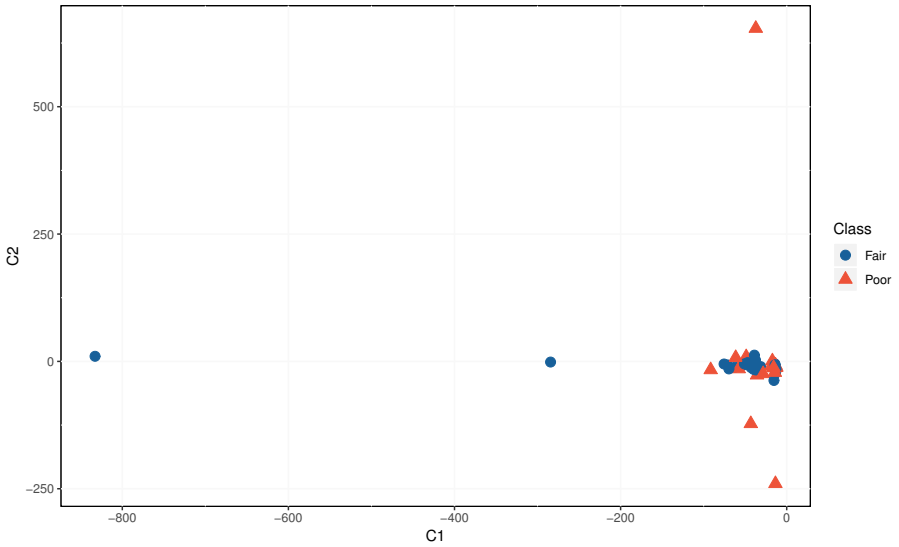


Figure 30: Clustering by PARAFAC using inverse kinematic relative variation scaled features and motion quality quantified using feature set 5

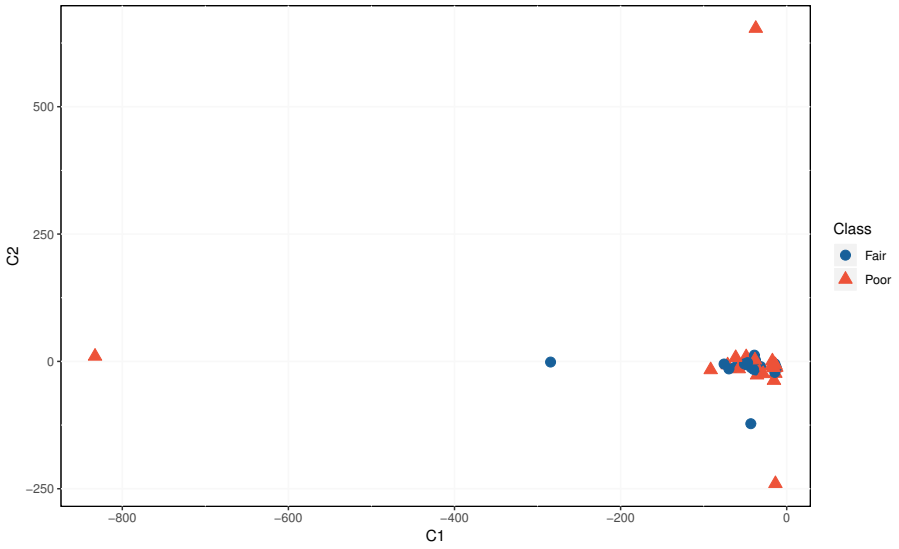


Figure 31: Clustering by PARAFAC using inverse kinematic relative variation scaled features and motion quality quantified using feature set 6

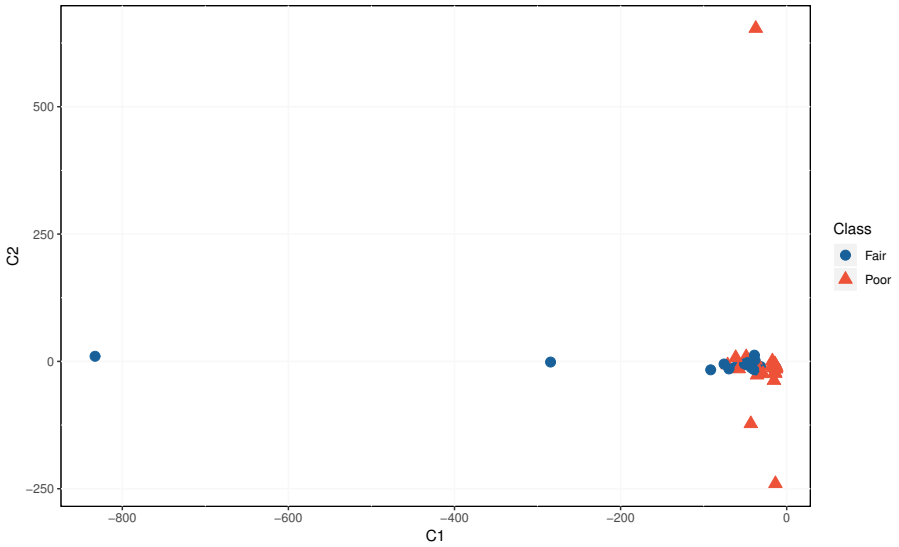


Figure 32: Clustering by PARAFAC using inverse kinematic relative variation scaled features and motion quality quantified using feature set 7

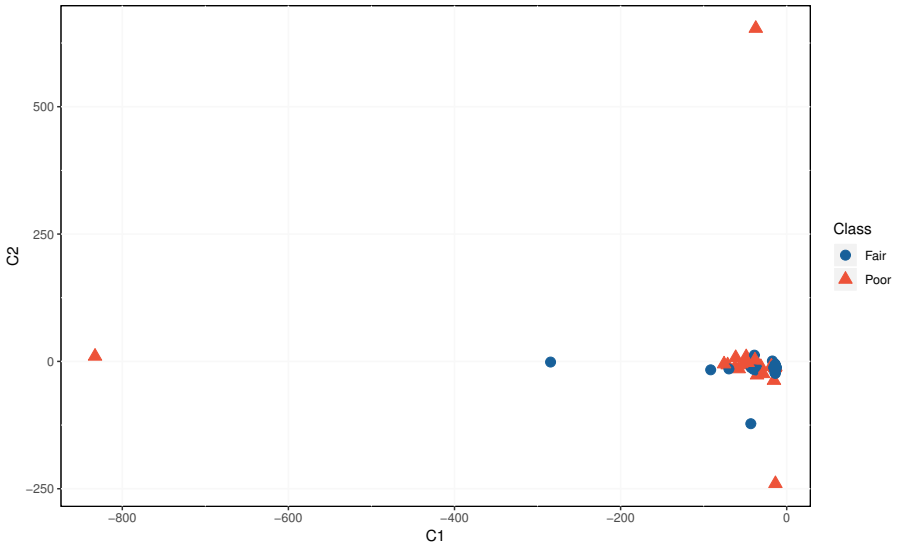


Figure 33: Clustering by PARAFAC using inverse kinematic relative variation scaled features and motion quality quantified using feature set 8

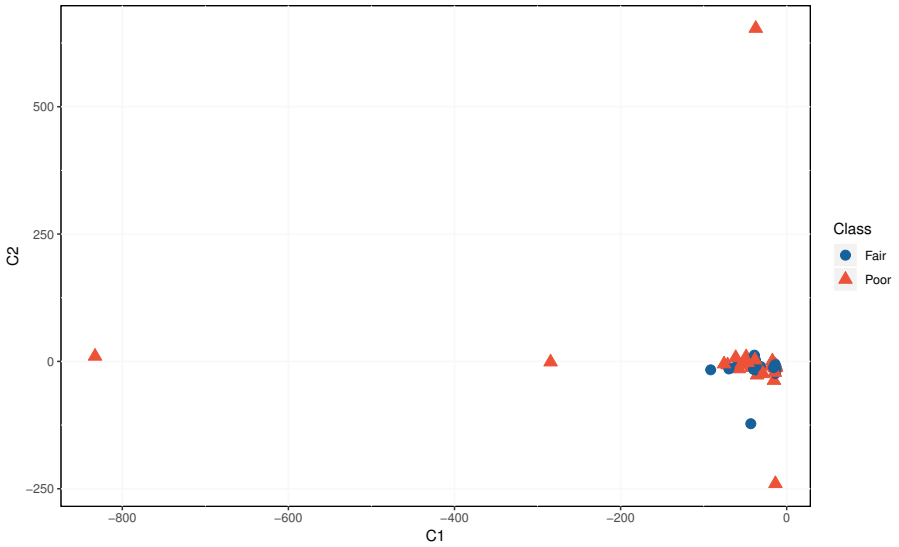


Figure 34: Clustering by PARAFAC using inverse kinematic relative variation scaled features and motion quality quantified using feature set 9