Predicting HIV Status Using Neural Networks and Demographic Factors

Taryn Nicole Ho Tim

A project report submitted to the Faculty of Engineering, University of the Witwatersrand, Johannesburg, in partial fulfilment of the requirements for the degree of Master of Science in Engineering.

Johannesburg, April 2006

Declaration

I declare that this project report is my own, unaided work, except where otherwise acknowledged. It is being submitted for the degree of Master of Science in Engineering in the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination in any other university.

Signed this ___ day of __________ 20___

____

Taryn Nicole Ho Tim.

Abstract

Demographic and medical history information obtained from annual South African antenatal surveys is used to estimate the risk of acquiring HIV. The estimation system consists of a classifier: a neural network trained to perform binary classification, using supervised learning with the survey data. The survey information contains discrete variables such as age, gravidity and parity, as well as the quantitative variables race and location, making up the input to the neural network. HIV status is the output. A multilayer perceptron with a logistic function is trained with a cross entropy error function, providing a probabilistic interpretation of the output. Predictive and classification performance is measured, and the sensitivity and specificity are illustrated on the Receiver Operating Characteristic. An auto-associative neural network is trained on complete datasets, and when presented with partial data, global optimisation methods are used to approximate the missing entries. The effect of the imputed data on the network prediction is investigated.

For my parents, family and friends, thank you for your understanding, patience and support.

Acknowledgements

I would like to thank my supervisor Prof. Tshilidzi Marwala for his valuable guidance, encouragement and support throughout the study. He not only proposed the project, but also sourced funding for it. Thanks to the Council for Scientific and Industrial Research (CSIR) and the National Research Foundation (NRF) for providing funding, which is much appreciated. The database used in this study originated from the Annual Antenatal Seroprevalence Surveys conducted by the Department of Health of South Africa.

Contents

List of Figures

List of Tables

List of Nomenclature

ANN	Artificial Neural Network
AUC	Area Under the Curve
CPU	Central Processing Unit
GA	Genetic Algorithm
HIV	Human Immunodeficiency Virus
HMC	Hybrid Monte Carlo
MAR	Missing at Random
MCAR	Missing Completely At Random
MCMC	Markov Chain Monte Carlo
MLP	Multilayer Perceptron
MSE	Mean Squared Error
PSO	Particle Swarm Optimisation
RBF	Radial Basis Function
ROC	Receiver Operating Characteristic

Chapter 1

Background

Acquired Immunodeficiency Syndrome (AIDS) was first defined in 1982 to describe the first cases of unusual immune system failure that were identified in the previous year. The Human Immunodeficiency Virus (HIV) was later identified as the cause of AIDS. Since the identification of the virus and the disease, very little has been effective in stopping the spread. AIDS is now an epidemic, which at the end of 2003 had claimed an estimated 2.9 million lives. Globally, an estimated 37.8 million people were living with HIV at the end of 2003, of which 4.8 million acquired the disease that year. The worst affected region is Sub-Saharan Africa, which has 10% of the world's population, but contains 70% of all people living with HIV. The epidemics have grown rapidly in all Southern African countries and South Africa has a high prevalence rate of 21.5% [1]. Of all countries in the world, South Africa has the highest prevalence of HIV/AIDS, followed by India.

Since AIDS removes people in the prime of their working and parenting lives from society, the economic impacts are enormous. Development and growth is adversely affected since productivity is lowered, incomes are diminished and poverty increases. Quantitatively, the World Bank calculates that AIDS may now be costing 24 African countries 0.5% to 1.2% of per capita growth each year [1]. All these factors in turn discourage investment exacerbating the situation, which affects government and business. Socially, health resources are stretched to capacity. The elderly and children are left to provide income for families, and education is no longer a priority.

To effectively manage this epidemic, accurate information on prevalence, improved understanding of the sociodemographic factors in which the epidemic occurs and the relative impact of interventions is required to construct and improve behaviour and treatment interventions. This is obtained by creating a model of the HIV epidemic. More specifically, the aim of this study is to predict the HIV status of an individual, given readily available demographic data. Ultimately, this knowledge will be used to construct health and social policies for HIV/AIDS prevention. Knowledge discovery and data mining in HIV related demographic are to be used to obtain the model. Artificial intelligence or machine learning techniques have been used successfully in medical informatics for decision making, and are used in this study for data mining.

To successfully perform knowledge discovery, an understanding of the data and application is necessary. Background information on knowledge discovery, HIV/AIDS and artificial intelligence techniques for data mining is presented in the following section; and the research question is defined in Section 1.5.

1.1 Knowledge Discovery

Knowledge discovery aims to uncover interesting, useful and novel patterns in data. Data is a set of facts, and a pattern is an expression in a language describing the facts in a subset of the facts. Data mining is the step in the knowledge discovery process when the data is actually searched through, using different algorithms. The steps of the processes are outlined below.

1.1.1 The Knowledge Discovery Process

These are the steps involved [2]:

1. Develop an understanding of the application, of the relevant prior knowledge, and of the end user's goals.
2. Create a target data set to be used for discovery.
3. Clean and preprocess data (including handling missing data fields, noise in the data, accounting for time series, and known changes).
4. Reduce the number of variables and find invariant representations of data if possible.
5. Choose the data-mining task (classification, regression, clustering, etc.).
6. Choose the data-mining algorithm.
7. Search for patterns of interest (this is the actual data mining).

8. Interpret the pattern mined. If necessary, iterate through any of steps 1 through 7.
9. Consolidate knowledge discovered and prepare a report

1.1.2 Data Mining

The general goal of data mining is to uncover relationships within the data and to predict outcomes. Data mining involves fitting models to or determining patterns from data. The general algorithm for data mining consists of three parts.

1. The model: The function of the model and its representational form, i.e. the data mining task
2. The preference criterion: The basis for which a model set of parameters is given preference for the particular data set. This is usually a goodness of fit function to the model. In terms of optimization, this can be seen as the objective function to judge the quality of the fitted models on observed data.
3. The search algorithm: the algorithm for finding particular models and parameters, given the data, model and preference criterion.

Data mining functions, used to create the model can be grouped under the following data mining tasks [3].

1. Exploratory Data Analysis (EDA): The goal of EDA is to explore the data without having any ideas of what is being searched for. This is typically done using interactive and visual techniques.
2. Descriptive Modelling: Descriptive modelling seeks to identify a model that describes all of the data or the process generating the data. Descriptions can describe different aspects of the data, such as the overall probability distribution (density estimation).
    - Clustering: Partitioning the data into several clusters, or groups of data based on similarity. The number of groups is determined by the data itself.
    - Segmentation: Grouping similar records into homogeneous groups. The number of groups is determined by the researcher, as is often used in marketing to define different customers.

- Dependency modelling: Models that describe significant dependencies between variables.
- Summarisation: provides a compact description for a subset of data.
- Sequence analysis: models sequential patterns like time series analysis in order to model the states of the process generating the sequence or to extract trends and deviation over time.

3. Predictive Modelling: The goal is to develop a model allowing one variable to be predicted from known values of other variables. If the predicted value is categorical, the function is classification, and if the predicted value is quantitative, the function is regression.

    - Classification: classifies the data into predefined categories: in this case, HIV positive and negative
    - Regression: maps a data item to a real valued prediction variable

4. Discovering Patterns and Rules: Discovering association rules describes relationship associations between different attributes, these types of rules indicate a statistical relationship between the attributes, but does not imply a causal effect. Rules are generated through the extraction of classification rules from the data.

5. Retrieval by content: The user has a pattern and wants to find similar patterns. The pattern could be an image or text for example, like searching for keywords in a web page.

Data Mining Algorithms

Numerous soft-computing algorithms exist for data mining, including: neurofuzzy computing, genetic algorithms, neural networks, rough sets, decision trees and hybridizations. The choice of algorithm is based on the data mining task, as different algorithms are better suited to different tasks.

1.2 HIV and AIDS

In order to identify the risk factors contributing to HIV infection, it is necessary to understand the disease, how it operates, is transmitted and diagnosed.

1.2.1 Definition

HIV is a retrovirus that infects cells of the immune system, such as CD4 cells and macrophages, and then destroys or impairs their function [1]. CD4 cells organise the body's overall immune response to foreign bodies and infections.

Immune deficiency arises from the progressive depletion of the immune system through this infection. Since the immune system is responsible for fighting off infection and cancers, cellular immune deficiency makes individuals more susceptible to opportunistic infections such as pneumocystis carinii pneumonia, toxoplasmosis, systemic and oesophageal candidiasis, generalized herpes zoster, cryptococcal meningitis, and to cancers such as Kaposi sarcoma.

1.2.2 Mode of operation and course of infection

First the virus penetrates the CD4 cells and copies the cells DNA to ensure that it cannot be identified and destroys the immune system. The virus replicates many times within the cell, and these new particles destroy the CD4 cell when they emerge. Each of the new viruses infects other cells. During the early stages of infection, many cells are infected and the number of virus particles in the body is high. The HIV status cannot be detected through tests, as insufficient antibodies have been formed, and this is called the window period. After this primary acute infection a prolonged period without obvious symptoms follows, later the body experiences severe immunodeficiency resulting in secondary opportunistic infections and cancers, the major causes of death in AIDS patients.

1.2.3 Methods of transmission

There are three main methods of transmission:

- Sexual contact
- Contaminated blood: sharing of contaminated needles in drug injection; sharing infected blood through blood transfusion
- Mother to child transmission during pregnancy, childbirth and breastfeeding

The predominant mode of transmission of HIV is sexual. According to [1], three factors influence the biological probability of transmission:

Type of sex: Anal intercourse carries a greater risk than vaginal intercourse, and receptive anal sex is more risky than insertive. The probability of HIV transmission is higher if there are lesions, such as would arise from rape and rough sex. The virus tends to be more easily transmitted from males to females, and the risk of male to female transmission is higher in girls younger than 16, compared to older women before the menopause. This higher biological vulnerability could be due to the immaturity of the genital tract and cervix. Risk of transmission of the virus through oral sex is small, but this is increased if there are abrasions in the mouth. Stage of illness: HIV infected individuals are more infectious during the earliest phase of infection before antibodies are produced, and at the later phase of the disease when the immune system is unable to combat the virus. Sexually transmitted disease: A person with an untreated sexually transmitted infection (STI) is, on average, six to 10 times more likely to pass on or acquire HIV during sex. An STI means there is more chance of broken skin or membranes allowing the virus to enter or leave the body.

1.2.4 Diagnosis of HIV and AIDS

Clinical Diagnosis

The clinical diagnosis of AIDS is difficult, as detection of HIV is limited in the early stages of the infection. Usually, a patient is suspected of AIDS when patients exhibit certain symptoms and suffer from opportunistic infections

Laboratory Diagnosis

It is possible to detect the HIV antigen (the virus itself) during the period when there are high levels of circulating virus particles, but the period is short, and the level of antigen declines until it is undetectable. Antibodies in the blood are usually detected as evidence of the virus, as they are cheaper and easier to detect than the virus itself. This can only be done by the current tests at the end of the window period. The two primary blood tests are the enzyme linked immunosorbent assay (ELISA) Test, and the Western blot assay used to confirm a positive ELISA test result.

The accuracy of diagnostic tests are measured according to sensitivity and specificity. High sensitivity indicates that the test is able to detect the presence of antibodies, i.e. minimising false-negative results. On the other hand, a test with high specificity, identifies all negatives correctly, producing no false positives. There is a margin of error, so the tests must be selected according to the different purposes, and used in different combinations.

1.2.5 Treatment

There is currently no treatment for eradicating the virus, but the existing treatments are able to control virus replication causing a reduction in HIV virus load in the blood. Such treatments are a combination of different antiretroviral drugs, known as Highly Active Antiretroviral Therapy (HAART).

1.2.6 Social and behavioural determinants of HIV

Circumstances such as poverty, low social status, inequality, gender discrimination, discrimination, marginalisation and criminalisation have been identified as increasing vulnerability to HIV infection [1]. These circumstances reduce or deny a persons access to HIV information, health services, means of prevention and support. Women in particular are a vulnerable population, in Africa, women are being infected at an earlier age than men, and more women are infected with HIV than men at all age levels, but especially in the group 15-24. In South Africa, two young women are infected to every young man [1]. The factors known to increase the risk of HIV infection among females in South Africa include the low social status of women and economic dependence on men, since these factors affect womens capacity to determine their sexual lives, with sexual decision-making being constrained by coercion and violence. Economic constraints may lead to prostitution or sex work [4].

1.3 Existing AIDS Models and HIV studies in South Africa

1.3.1 HIV statistics

Although HIV statistics should ideally be collected via community based serosurveys, only a few of such surveys have been carried out, due to financial, logistic and ethical considerations [5]. Many countries prefer sentinel surveillance systems that monitor the population anonymously over time and the most common sentinel is pregnant women [5]. Thus, the most common source of HIV statistics is from antenatal clinics.

Reasons for this method are that antenatal clinics are found in most parts of the world, so they provide a common basis on which to compare regional and national statistics [6]. Two main measures of HIV are HIV incidence and prevalence. The HIV incidence rate is the percentage of people who are uninfected at the beginning of the period who will become infected over the twelve months [7]. Incidence is thus the number of new infections, and can be used as a measure of the effectiveness of preventative strategies, like education. HIV prevalence is given as a percentage of a population. It is the proportional number of people currently infected with HIV. This figure often is not a true reflection of the situation, since in developing countries prevalence is a measure of infection among people who have access to health care, such as antenatal clinics. Reported figures indicate an actual positive result, but it is difficult to deduce trends over time as there is often a lag in the collection of figures at a central agency. Estimates vary depending on the statistical methods used to produce them, as well as the data on which they are based.

1.3.2 National Antenatal and Syphilis sero-prevalence surveys

In South Africa, antenatal and seroprevalance studies have been conducted annually since 1990 to measure HIV and syphilis prevalence: they form the main source of data for tracking the progression of the HIV/AIDS epidemic in the country [8] [9]. The objective of the survey is to estimate HIV and syphilis prevalence among pregnant women attending public sector antenatal clinics and describe HIV and syphilis trends in terms of time, place (province) and age among pregnant women. The data can thus be used to track the epidemic in the different provinces of South Africa. Limitations of the study are that only certain sites were selected for participation in the study. A second limitation of the study is that the use of public sector health care facilities excludes women who utilise private health care facilities. The most recent survey was conducted in October 2003 by the Department of Health of South Africa [9]. Quality assurance is the responsibility of the National Institute for Communicable Diseases (NICD) of South Africa performing external quality control of the HIV prevalence testing.

The survey is conducted every year during October, and all pregnant women attending antenatal care for the first time during the current pregnancy are eligible for inclusion in the survey. Participation in the survey is voluntary and the study is anonymous, cross-sectional and unlinked. Selected sites in all nine provinces of the country participate in the study. Demographic details such as race, age, education, gravidity, parity and fathers age are collected at the clinics, and a blood sample is taken. Each sample is tested for HIV prevalence using one ELISA at participating laboratories, Rapid Plasma Reagin (RPR) [10] is used to test for syphilis. Samples that test positive for prevalence are then tested for incidence. All the results in the province are sent to the provincial co-ordinator for second data entry and verification. Provincial data is then sent to the Epidemiology directorate, where it is re-checked, cleaned and merged into a single [9] database.

1.3.3 HIV models

The following organisations have developed models: Statistics South Africa, Actuarial Society of South Africa (ASSA), UNAIDS, Human Sciences Research Council and the Department of Health. There is a wide range of estimates for both the number of people infected with HIV in 2004 [7], and the HIV prevalence rate in the total population of 2004, as seen in Figures 1.1 and 1.2.

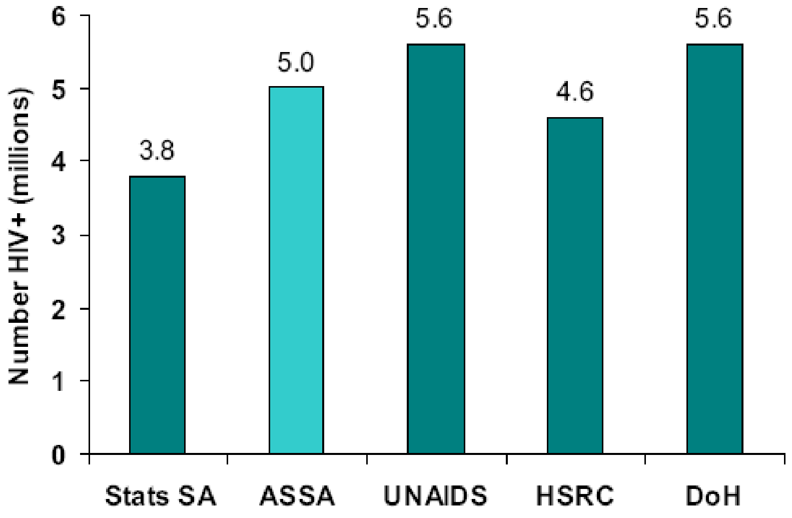

Figure 1.1: Number of people infected in 2004

The models differ because of the sources of data used to construct the models as well as the assumptions made. Two major problems are the differences in projected population which affects the prevalence, and the generalisation by projection of antenatal survey data onto the general population. A method for adjusting is presented in [5].

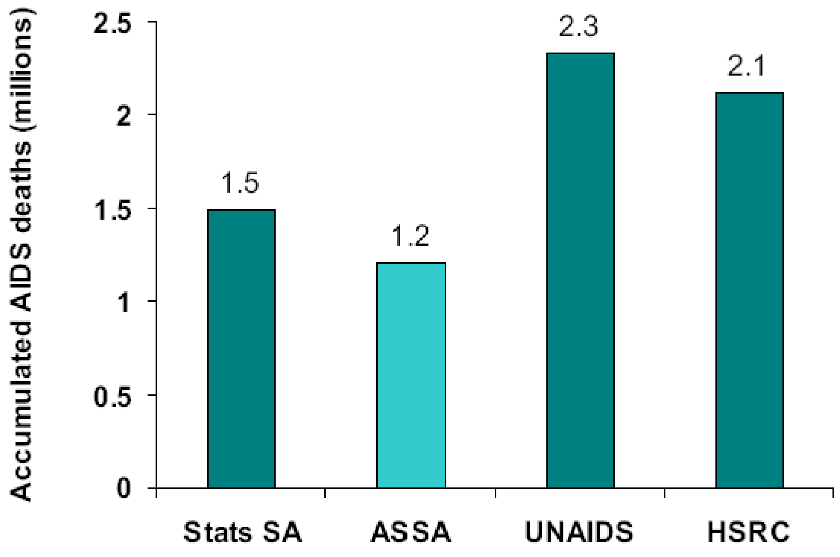

Figure 1.2: HIV prevalence rate in total population in 2004

1.3.4 Demographic, behavioural and social risk factors for HIV

A few national population based surveys have been conducted to determine behavioural and social factors influencing the prevalence of HIV in South Africa. These are: the South African African Health Inequalities Survey (SAHIS 1994), the South African Demographic and Health Survey (SADHS) (Department of Health 1998b), and the Human Sciences Research Councils (HSRC) surveys (1997, 1999, 2001) [4]. Behavioural factors such as condom use is related to different demographics, but these studies do not examine the relationship between behaviours and HIV sero-status.

The most comprehensive population-based survey is the Nelson Mandela/HSRC Study of HIV/AIDS carried out in 2002, which provides more accurate HIV-related sexual behaviour risk profiles. The effects of race, age, locality, province, history of diagnosis of STI, and education levels are investigated. The national prevalence was significantly higher among Africans in years 2000 - 2003 than any other race groups. Free State, Gauteng and Mpumalanga have the highest HIV prevalence in South Africa. People living in informal urban areas were significantly more likely to be HIV positive than those living in urban formal areas. Those living in urban formal areas had a significantly higher prevalence when compared to residents of farm and tribal areas. There is no simple relationship between HIV infection and levels of education. School attendance may increase access to both information and potentially to prevention interventions. However, the improvements in socio-economic status and lifestyle changes that go with higher educational attainment may be associated with behaviours that increase the risk of HIV infection [4].

International studies have identified key behavioural factors such as age, condom usage, median age at first sex, and knowledge about AIDS [11]. Risk factors influencing prevalence are related to the risk behaviour of the male partner: marital status, having multiple sexual partners, having a male sexual partner who drank alcohol or who had higher income [12; 13].

1.3.5 Predicting HIV status from non-serological information

Previous studies, [14; 15] suggest that it is infeasible to predict HIV status using non-serological information.

Location	Year	Model	Factors	Percentage Accuracy	
				True Positives	False Negatives
Kenya [14]	2000	Poisson Regression	Anaemia, malarial paristaemia, history of being treated for vaginal discharge, fever, alcohol consumption	74	52
Congo [15]	1992	Logistic Regression	age, history of blood transfusion and/or hospitalization, district of residence, duration of the relationship, number of living children, and number of deceased children.	80	50

Table 1.1: Results from HIV prediction studies using non-serological information

A study in Zaire [16] performed on 6312 women between the ages of 15 and 45 years using a combination of factors modelled by logistic regression proved that only a model which included indicators of present illness with AIDS/HIV-related symptoms (chronic fever, diarrhoea, or profound weight loss) was predictive of HIV serostatus.

1.4 Neural Networks as statistical tools for medical research

This section introduces the use of neural networks in the medical field in general, as well as their suitability for classification and regression. More detailed information regarding the structure and functioning of neural networks is presented in Chapter 2.

Artificial intelligence has been used in medicine for the clinical functions of diagnosis, prognosis and survival analysis, and decision support. It has been used in a wide variety of medical domains such as oncology, critical care, tuberculosis, cardiovascular and renal transplantation. Artificial Neural Networks (ANNs) perform well in pattern recognition, and are suitable for signal processing (EEG, ECG, and haemodynamic signals), as well as image processing (mammography, chest radiographs, tomography, nuclear medicine imaging, magnetic resonance). A common task in medicine is thus classification using predictive models.

1.4.1 Neural networks versus polynomial regression

Nonlinear regression requires a polynomial function to approximate the model and a major difficulty is selecting the order of the polynomial (often done by guessing). As the number of independent variables increases, the number of possible regression models becomes intractable. Linear models are easy to construct in linear regression but training takes a long time, and offers no advantage over the calculation by Fishers linear discriminant, or other statistical methods. The interpretability of neural networks has often been criticised since none of the coefficients can be interpreted as can be done with regression models.

1.4.2 Neural Networks for classification and regression

The major strength of neural networks in modelling multidimensional spaces, is their ability to scale with increasing dimensionality of the data. The ability of a polynomial to model a non-linear function is limited by the number of terms in it, and hence the order of the polynomial allows it to model higher dimensions. To accurately determine the numerous parameters that would arise in a high order polynomial, large data sets would be required. In contrast to polynomials, which use one function to model the relationship, neural networks use superposition of many functions of a single variable each. These functions are known as the hidden functions, and adapt as the complexity of the model grows, not simply with dimensionality.

Additionally, for the case of classification, the outputs of a neural network can be interpreted as posterior probabilities, provided the error function used to train the network is chosen appropriately. This property is explained in detail in Section 4.2 Chapter 4.

1.4.3 Alternatives to neural networks for classification

Tree models

The space spanned by the input variables is partitioned recursively to maximise a measure of class purity. Although they are very flexible and can handle mixed variables, their sequential nature can lead to suboptimal partitions of the space of the input variable.

Support Vector Machines

Support vector machines implement a linear decision surface that separates data in an extended measurement space. This linear decision surface is equivalent to a nonlinear decision surface that separates the data in the original measurement space [3]. The function used to separate the two classes is called the margin, and it attempts to optimise the location of the linear decision boundary such that the best possible generalisation performance is obtained. This is an optimisation problem which can be slow, as it requires O(n2) storage and O(n3) time to solve [3].

Artificial intelligence methods in AIDS research

Despite the numerous applications of artificial neural networks to classification in medicine, very little attention has been made to the HIV/AIDS prevention and planning [17]. Artificial neural networks have been used to classify and predict the symptomatic status of HIV/AIDS patients ˜citeLee:2001pn. The data used were all the complete entries from a publicly available AIDS Cost and Services Utilization Survey performed in the U.S. A multilayer perceptron, with 15 linear inputs and 3 hidden logistic nodes and one output, was trained using 200 epochs with a learning rate of 0.1 and momentum of 0.1. 1026 cases were used for training and 667 HIV cases were used for testing. The inputs are: sex, race, exposure rate (homosexual, IV drug user, heterosexual), medical records (total number of patient admission, total number of inpatient nights, total number of ambulatory visits, total number of emergency room visits, total number of hospital clinic visits, total number of private physician visits). The output is HIV status or AIDS status. The best accuracy obtained was 587 correct (88%).

A study was performed to predict the functional health status of HIV and AIDS patients defined as well or not well, using neural networks [18]. The inputs were medical care access, such as number of emergency room visits and inpatient nights.

Most other applications of neural networks in AIDS research are in bioinformatics pertaining to modelling of the virus on a molecular level, such as the prediction of HIV-1 Protease Cleavage Sites .

1.5 Problem statement

A classification system is to be developed using neural networks. This involves the selection of relevant parameters and optimisation of the neural network architecture to be able to classify an outcome in the given data. The application selected for this study is HIV classification using demographic factors, and the developed algorithm should be applicable to any complex system with sufficient data.

1.5.1 Objectives

Classify HIV status

The aim of the study is disease risk analysis, and the primary objective is to use artificial intelligence methods, namely, neural networks to perform knowledge discovery and data mining on HIV clinical and demographic data, resulting in a classifier of HIV status of a patient based on demographic inputs.

Incomplete data

A major problem with using neural networks is the requirement for a complete set of data. Medical data often contains many missing entries, and the secondary objective of the study is to replace the missing data. A method for missing data imputation was developed by Abdella and Marwala [19], and this is to be used to complete the data set. The effect of missing data (number of entries missing, the nature of the variables that are missing) on the prediction accuracy was investigated.

1.6 Overview of Approach

For the classifier, a multilayer perceptron was trained using the Bayesian framework, and a Genetic Algorithm was used to obtain the neural network architecture parameters. The missing data optimiser consists of an auto-associative neural network and the problem of minimising the error between the initial guess and output was optimised with Particle Swarm Optimisation.

1.7 Organisation of the Remainder of the Report

Background and general theory necessary to understand the design of the system is covered in Chapters 2 and 3. As mentioned in the previous section, there are two distinct parts to the study: classification and approximation of missing data. Both parts are performed by neural networks. A background on Artificial Neural Networks is presented in Chapter 2, with an explanation of the design and training of the different types of neural network architecture. This work focuses on the use of Multilayer Perceptrons, as they proved to be the most successful architecture.

Stochastic optimisation methods were used in each part of the study in different contexts: firstly to select the parameters for the design of the classifier, and secondly, in the actual implementation of the missing data estimator. The methods implemented are Genetic Algorithms and Particle Swarm Optimisation, detailed in Chapter 3.

The actual design and reasoning for the approach taken follows in Chapters 4 and 5. The theory behind using neural networks for classification and their probabilistic interpretation is given in Chapter 4. Chapter 5 contains the design of the missing data system, with an investigation into different optimisation methods. Implementation and the results obtained for both parts is presented in Chapter 6. This is followed by a review and conclusion in Chapter 8.

Chapter 2

Artificial Neural Networks

The artificial neural network (ANN) originated as a model for biological neural networks. Neural networks are viewed as a computational method of representing the non-linear functional mapping of an input to an output in this paper. In classification, they are function approximators, where the function approximated is the probabilities of membership to classes as a function of the input variables. Neural networks are thus used as predictive data models.

2.1 Overview

The functional performance of the network depends on three factors: the selection of inputs, the network structure and the training of the network. The design and training of the network involves presenting it with a set of corresponding inputs and outputs, and training it to adapt to provide the outputs given a specific input.

2.1.1 Selection of Inputs

The number and type of input nodes is determined based on the nature of the problem and the data available. Inputs may be either binary, or continuous. There should be sufficient input nodes to represent the information relevant to the problem. The same number and type of inputs are used for both architectures, as their performance is compared.

2.1.2 Network Structure

Although there are many different ANN models, they all have the same basic structure. An ANN is a network of many simple processors (”neurons” or nodes), each with an associated memory. The nodes are connected in layers, most commonly three layers: an input layer, a hidden layer, and an output layer. The networks considered in this study are feed-forward: outputs of nodes in each layer are connected to nodes in the next successive layer. No information is fed back from the outputs to the input layers. Additionally, the networks may be fully connected: all nodes are connected to all other nodes.

2.1.3 Artificial neurons

Each node receives information from many other nodes, and the multiple signals are combined to compute the single output associated with each node. Memory is implemented by way of the connections that connect the neurons to each other, and these carry numeric information. Each neuron receives data from another neuron or the information source, processes the data, and the output is then sent to the next neuron. This result may then be processed by the next neuron. In this way, neural networks are a parallel computational method. Due to the nature of the structure, the internal components are parallel, thus multiple neural networks can be combined in the same fashion.

2.1.4 Training

The major strength of neural networks is that they are able to model data, mapping the interconnections between the given inputs and outputs so that it is unnecessary to have knowledge of the exact relationship. The form of the mapping is governed by the number of adjustable parameters. Neural networks learn by example, and if trained properly, are able to generalise and extend the knowledge to samples that are unseen. Training is the process of adjusting the weights of the connections according to the data. This adjustment can either take place after one example is presented to the network (online training), or after the entire set of examples has been presented to the network (batch training). The training method depends greatly on the structure of the network and radial basis functions are trained in a very different way from the way multilayer perceptrons are trained.

2.2 Multilayer perceptrons

The most widely used neural network architecture is the multilayer perceptron. The multilayer perceptron structure provides a non-linear mapping from a real-valued input vector x to a real valued vector y. It can thus be used as a non-linear model for both regression as well as classification, depending on the interpretation of the output(s). The functionality of the MLP is based on its multi-layer structure and activation functions. The main idea in MLPs is that the input vector is successively modified through multiplication by weight matrices in the different layers, and the products are transformed by non-linear activation functions.

2.2.1 Network Structure

The architecture of the MLP consists of an input layer, hidden layers and an output layer, and this is illustrated in Figure 2.1. It has been proven that a single hidden

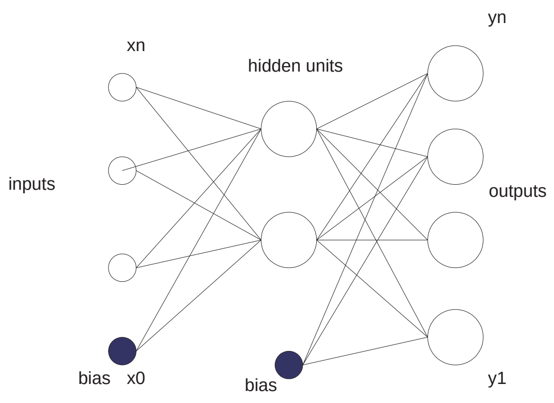

Figure 2.1: Multilayer Perceptron

layer feed-forward neural network is capable of approximating uniformly any continuous multivariate function to any desired degree of accuracy, provided that the number of hidden units is sufficiently large [20; 21]. A network with no hidden layer, that is, with only one layer of nodes with activation functions is sufficient for two class classification, only if the data points are linearly separable. This network is unfeasible, as for a set of N points in d dimensional space, a network with N/d nodes is needed to correctly separate the points into two classes [21].

Connections and weights

The inputs are fed directly into the input layer, and each input is connected to every node in the hidden layer via a feed forward connection. The output from each hidden node is connected to every output node. Shortcut connections can exist, but in this study they are not considered. Connections that connect one layer to the preceding layer are only found in recurrent neural networks, where the outputs are fed back as inputs. Each connection has a a weight which is a scalar value that is multiplied by the signal at the origin of the connection.

Neurons

Consider one neuron. The signal this neuron receives consists of the inputs directly connected to the neuron, and each of these inputs is multiplied by the weight connecting it to the neuron. These products are then summed to form the activation at the neuron, which will either cause the neuron to fire, or produce no signal. This is achieved mathematically with an activation function. Non-linearity is introduced to the system through the activation function, which is non-linear.

2.2.2 Mathematical Function

Every hidden node has an associated activation due to its weighted connections to each input x_i. There is also a bias b_j at each hidden node, thus the first layer activations are defined in [21] by:

$$a_j = \sum_{i=1}^{N} w_{ij}^{(1)} x_i + b_j^{(1)} \tag{2.1}$$

Where i is the number of inputs and j is the number of hidden nodes.

The significance of the number of hidden nodes in a classifier is that each hidden unit divides the input space with a hyperplane, so that activation $z = 1$ is on one side of the hyperplane, and $z = 0$.

These activations are transformed by a non-linear activation function in the hidden layer. There are many possible choices for non-linear activation functions, such as threshold functions and linear functions, but in this study sigmoidal units are used. Sigmoidal functions are S-shaped, mapping the interval $(-\infty, \infty)$ onto the interval $(-1, 1)$) for hyperbolic tangent functions, and onto the interval $(0, 1)$ for the logistic sigmoid function. They are differentiable, which is a necessary property for back-propagation, and are able to represent smooth mappings between continuous variables. These types of functions are used as activation functions as they are monotonic. In the hidden layer, hyperbolic tangent functions are used as they are equivalent to the logistic function through a linear transformation, but in addition to all the properties offered by the logistic function, also give rise to a faster convergence of training algorithms than the logistic functions [21]. The outputs of the hidden nodes z_j are thus:

$$z_j = \tanh(a_j^{(1)}) \tag{2.2}$$

For a two class classifier one output node is sufficient, so there is only one activation at the second layer. The outputs from the hidden layer are connected via weighted connections to the output node and biased to form the second layer activation [21]:

$$a^{(2)} = \sum_{j=1}^{h} w_j^{(2)} z_j + b^{(2)} \tag{2.3}$$

This second layer activation is transformed by the logistic output activation function, as it operates in the range 0 to 1, and it allows the output to be given a probabilistic interpretation, since it is derived using Bayes theorem to represent the posterior probabilities of membership to classes. Additionally, the sigmoidal function is able to represent both non-linear functions as well as linear functions (if $|a|$ is small).

$$y \equiv \frac{1}{1 + e^{-a^{(2)}}} \tag{2.4}$$

The output y, is a continuous scalar bounded between 0 and 1, thus to use y as the indicator of class membership it needs to be converted to binary values using a threshold.

Since the resultant model is non-linear, when applied to classification, the decision boundary between the classes produced by the network is also non-linear. This is an advantage over most other classification methods such as trees which have linear decision boundaries. Non-linearity allows for highly flexible decision surface shapes, but since non-linear estimation of the parameters is not straightforward iterative techniques are used (training). The process of obtaining the weights that produce the model is called training.

2.2.3 Training

The weights and biases that result in the optimal decision surface are determined through the minimisation of the error produced by the network. The error function is the sum of squares error in the target outputs t, and those produced by the network y. The optimisation methods used are gradient based, and these require the derivative of the error function with respect to the weights. Sigmoidal activation functions are used because they are differentiable, and this is a necessary requirement in error back-propagation. The error back-propagation algorithm solves for the weights through the propagation of the errors backwards throughout the network. Training begins with the random initialisation of the weights and biases, and these values are adjusted until the error converges, or until a certain number of training cycles has been exceeded. Since the target outputs are used as a reference (in the calculation of error), this method of learning is referred to as *supervised learning.*

2.2.4 Decision Surface

The output activation function determines the decision surface in the input plane, in this case a logistic output activation function is used, since the range is 0 to 1, a typical value for a threshold would be 0.5, and thus y = 0.5 defines a decision surface in the output space that can separate the classes. Every node in the hidden layer defines a decision plane (or hyperplane) in the input space, and the hidden nodes combine to form a non-linear decision surface. A network with only hidden layer cannot generate arbitrary decision boundaries, however any given boundary can be approximated arbitrarily closely if the activation functions are sigmoidal.

2.2.5 Generalisation

The aim of the network is to capture the statistical properties of the data to be able to make accurate predictions for new inputs, that is, to generalise. Poor generalisation can either arise if the network is not sufficiently complex and is unable to accurately represent the process, or it may be too flexible, and be fitted to the noise of the training data, or the data is too difficult to be mapped using neural networks. There needs to be an optimisation between these two points, to find a model that represents the data accurately enough to correctly predict unseen data, and this is done by controlling the complexity of the model.

The first method is to control the complexity through the structure of the network. A simple model has fewer parameters that can be adjusted. In multilayer perceptrons, the number of hidden nodes is proportional to the complexity of the network, since the more nodes there are, the more weights and biases there are to adjust. Reducing complexity equates to reducing the number of hidden nodes, and this is done either by starting with a small number of nodes and adding more, or by pruning the network of hidden nodes to a smaller number of them.

The second method is regularisation, which aims to have smoother network mappings by adding a penalty to the error function. A weight decay can be added to the error function so that the weights are encouraged to be small as training progresses. Small values of weights mean that the multilayer perceptron represents a linear mapping, since these values correspond to the central region on the sigmoidal function. Early stopping is an alternative approach to weight decay: it ensures that the network does not over-train, and prevents over-fitting to the training data. In this method, cross validation is employed: the network is trained on a training data-set, and the error on another set, the validation set is compared to this error in training data. When the validation error starts to increase, the network has become over-trained, and training is stopped.

2.2.6 Radial Basis Function

The major difference between radial basis function networks and MLPs is that the activation of the hidden unit is determined by the *distance* between the input vector and a weight vector. Their major attractive feature is that they train much faster than MLPs, and this is attributed to the two stage training procedure.

Network Structure

The structure of the radial basis function network is similar to that of the MLP, however instead of hidden nodes with activation functions, the RBF has basis functions. Also, there is usually only one hidden layer in an RBF network. Like the MLP, the nodes are feed-forward, fully connected and have weights and biases.

Mathematical Function

Radial basis functions use a combination of supervised and unsupervised learning techniques. Learning in the hidden layer is unsupervised, using methods like k-means clustering. These clusters are used as the starting points from which supervised learning in the outer layer takes place. A least mean squares method is used for this stage of training.

There are several types of basis functions, but the most common is the Gaussian

$$\phi(x) = exp\Big(-\frac{x^2}{2\sigma^2}\Big) \tag{2.5}$$

where σ controls the smoothness of the properties of the interpolating function. The parameter σ is the width parameter, and the value for it is determined in training. This function is localised, as $|x| \rightarrow \infty$, $\phi \rightarrow 0$

The thin plate spline function has also been used, and can be written as follows:

$$\phi(x = x^{2ln(x)}) \tag{2.6}$$

The output is represented as the sum of the product of the basis functions with the biases added:

$$y(x) = \sum_{j=1}^{M} w_j \phi_j(x) + w_{10} \tag{2.7}$$

In a similar way that the biases were absorbed in the MLP as another input fixed at 1, the biases can be absorbed into the summation as another basis function with the activation set to 1.

The basis function is

$$\phi_j(x) = exp\Big(-\frac{|x-\mu_j|^2}{2\sigma_j^2}\Big) \tag{2.8}$$

where x is the d-dimensional input vector, μ_j is the vector determining the centre of the basis function ϕ_j

Decision Surface

The activations of the basis functions can be interpreted as the posterior probabilities of the presence of the corresponding features in the input space, and the weights can be interpreted as the posterior probabilities of class membership, given the presence of features. Instead of the hyper-planes in the MLP approach to classification, there are hyper-spheres separating the classes in the RBF.

Training

Training an RBF takes part in two stages. The first stage is an unsupervised learning procedure, where the input data set is used to determine the parameters of the basis functions. These parameters are the basis function centres μ_j, and the widths of the basis functions σ_j. Only the input data is used, and no target information is used. The parameters are set such that the network models the unconditional data density. Selecting the centres can be done in a few ways: the simplest is to randomly select data points as the basis function centres. In this study, a more advanced method of clustering the data, and using the clusters as centres was implemented. An Expectation Maximisation algorithm was used to train a Gaussian mixture model.

In the second stage, with the basis functions fixed, supervised learning takes place, to optimise the second layer of weights. This optimisation is a quadratic problem, and the values are solved for using linear algebra.

It is possible to train a RBF network using a single stage supervised learning method using non-linear optimisation algorithms as is done in training of MLPs. To use this method, the error as well as the derivative of the error function are required, and these are typically computationally expensive.

Generalisation

In a similar way that limiting complexity improves generalisation in MLPs, the network complexity is used to improve generalisation in RBFs. A reduced number of hidden nodes, or in this case, basis functions, limits the networks ability to model the data exactly, and a smoother mapping is achieved

2.3 Bayesian Regularisation Training for Neural Networks

2.3.1 Background

The most difficult and time consuming aspect of neural network design is the model complexity. The optimal number of degrees of freedom is controlled by the architecture in the number of hidden nodes, and modelling ability depends on the training data (the number of training samples, as well as the noise in the data) [22]. Standard neural network techniques use an experimental method to compare performance on the validation set to optimise model complexity, which is computationally intensive. Since all parameters affect the modelling ability of the network, the effect of any single parameter cannot be identified as each individual effect cannot be isolated. The presence of the validation set also reduces the amount of data available for training and testing.

The aim of neural network training is to adjust the weights such that when presented with an unseen set of inputs characteristic of the underlying function, the neural network will be able to predict the output. Conventional neural network training produces a single set of weights, according to maximum likelihood, to make predictions. Instead of a particular weight vector, the Bayesian training method produces a posterior distribution over network weights, and when a set of inputs is applied to the network, a distribution over the outputs is produced. This predictive distribution is informative of how uncertain the prediction is.

The Bayesian inference system handles the unknown degree of complexity by defining non-informative prior probability distributions that determine model complexity, and the resultant model is an average of all model complexities weighted by the posterior probability given the data sample [22]. The Bayesian framework allows the model to have differing complexity in different parts of the model, by the grouping of parameters to common hyperparameters. Analysis of regression results is possible in Bayesian techniques with the posterior predictive distributions from which confidence intervals can be calculated, and in classification, overconfident predictions in areas of sparse data are avoided [23].

There are two approaches for selecting the network architecture. Neal argues that there is no statistical need to limit the complexity of the network architecture in the Bayesian implementation, so the approach is to use a general architecture with many hidden units (in several layers or groups) controlled by hyperparameters [24]. In contrast, MacKay uses an evidence framework to choose between architectures, which obeys Occam's principle. The issue of model complexity is embodied in Occam's Razor, the principle that states that simple models should be favoured over complex ones [25]. The Bayesian evidence framework consists of the aspects model comparison and selection, feature selection and the use of multiple neural networks, and in this paper, it is this approach that is considered for the purpose of classification with neural networks.

2.3.2 Probability Theory

Bayes Theory is used to calculate a conditional probability $P(A|B,C)$ which is the probability of A given the probability of B and C. The conditional probability is also related to the joint probability of A and B:

$$P(A|B,C) = \frac{P(A,B|C)}{P(B|C)} \tag{2.9}$$

The conditional probability does not mean that B and C are a prerequisite for A to occur, instead, it is a measure of the probability A will happen, if B and C also occur. For example, if A is the probability that a person has a disease, and B is the person's age, the $P(A|B,C)$ is a number between 0 and 1 describing the chance that the person has a disease, taking into account their age, and the model for the disease and risk factors (C). When the extra information changes, we can adjust the conditional probability according to Bayes theorem:

$$P(A|B,C) = \frac{P(B|A,C)P(A|C)}{P(B|C)} \tag{2.10}$$

In the context of learning from the data, the prior distribution is the probability distribution over the network weights that define the assumed pattern in the data being learned. After having observed the data, the opinion of the probability distribution over the weights will change to the posterior distribution. This means that weights that do not represent the data well will have a much lower probability, while those that fit the data well will have an increase probability.

Model selection

Conventional neural network learning starts with an initial weight vector w_0, and results in a single weight vector w in weight space, close to the point of maximum likelihood, or minimum error. This method is highly dependant on the training data set, and any slight changes in this set or changes in the initialisation of the weights results in a different optimum weight vector. In contrast, the uncertainty in the value of weight vector values is taken into account in the Bayesian learning process. An ensemble of possible solutions is considered, instead of simply considering one best solution, and this is done using Bayes theorem and the conditional probability. There are two levels of inference in model comparison, firstly, we assume that a particular model is true, and infer values for the weights, given the data [26].

$$P(w|t,x,H) = \frac{P(t|w,x,H)P(w|H)}{P(t,x,H)} \tag{2.11}$$

This is equivalent to

$$PosteriorDistribution = \frac{Likelihood \times PriorDistribution}{Evidence} \tag{2.12}$$

The prior distribution over weight space is transformed to the posterior distribution using the likelihood and evidence of the model, and with increasing training data, the posterior ensemble becomes concentrated around the most probable value of w. The prior distribution in this case, is the probability of the weights, given the model. The maximum of the posterior is the weight vector with the maximum probability, and this can be found with gradient methods [26]. Error bars and confidence intervals can be approximated from the curvature of the posterior distribution, and more details can be found in [26].

The second level of inference determines which is the most probable model given the data, by obtaining the posterior probabilities of each model, as seen in Equation 2.13 [23].

$$P(H_i|D) = \frac{P(D|H_i)P(H_i)}{P(D)} \tag{2.13}$$

where H_i is a model, and D is the data, consisting of w and t. We initially have no information to differentiate one model from another, so $p(H_i)$ is the same for all models. The normalising constant, $P(D)$ is independent of the models, so the model with the highest posterior distribution is the model with the highest evidence (the probability of the data given the model), and this quantity can be used to compare different models.

The evidence is evaluated using Equation 2.14 [26]

$$p(D|H_i) = \int p(D|w, H_i)P(w|H_i)dw \tag{2.14}$$

Training and regularisation

The previous sections discussed the model selection, and adjusting of weights without details of the prior weight distribution.

$$p(w) = \frac{1}{Z_w(\alpha)} exp(-\alpha E_w) \tag{2.15}$$

The purpose of Z_w is to normalise the probability such that:

$$\int p(w)dw = 1 \tag{2.16}$$

thus

$$Z_w = \int exp(-\alpha E_w)dw \tag{2.17}$$

$$E_w = \frac{1}{2}|w|^2 = \frac{1}{2}\sum_{i=1}^{W} w_i^2 \tag{2.18}$$

where W is the total number of weights and biases.

The advantage of the Gaussian prior, is that it simplifies analysis, and the normalisation coefficient is just

$$Z_w(\alpha) = \left(\frac{2\pi}{\alpha}\right)^{W/2} \tag{2.19}$$

Since α controls both weights and biases, it is termed a hyperparameter.

To define the *likelihood* function, a model for the distribution of target values for a given input vector must be defined. In binary classification, the output y represents the posterior probability $P(C_1|x)$, and the posterior probability of the other class is $P(C_2|x) = 1 - y$. For regression, a Gaussian noise model with zero mean and constant inverse variance β is appropriate, but instead, for classification, a Bernoulli random variable is used for the distribution. In classification, there is no equivalent β hyperparameter. The posterior distribution is

$$p(D|w) = \prod_n y(x^n)^{t^n}(1 - y(x^n))^{1-t^n} = \exp(-G(D|w)) \tag{2.20}$$

where G is the cross entropy error function, given by

$$G(D|w) = -\sum_n t^n lny(x^n) + (1 - t^n)ln(1 - y(x^n)) \tag{2.21}$$

The distribution is normalised since the targets have values 0 or 1, and the normalisation integral is a sum of terms:

$$exp(lny) + exp(ln(1 - y)) = y + (1 - y) = 1 \tag{2.22}$$

$$E_D(D|w, x) = -\sum_{n=1}^{N} t^n lny(x^n, w) + (1 - t^n)ln(1 - y(x^n, w)) \tag{2.23}$$

The normalisation factor is the integral of $exp(\beta E_D)$ as in 2.17.

The total error function is:

$$E = S(w) = E_D + \alpha E_w \tag{2.24}$$

The Gaussian prior distribution for the weights thus results in a cross entropy error function with a weight decay regularisation term. Keeping α constant, as N the number of training samples increases, the first term becomes more dominant and the second term insignificant. Hence, the maximum likelihood solution is a special case of the Bayesian most probable weight vector, and is simply an approximation.

Substituting the likelihood and prior distributions into Bayes theorem equation, we obtain the posterior distribution of weight values.

$$p(w|D) = \frac{1}{Z_s} exp(-G - \alpha EW) = \frac{1}{Z_S} exp(-S(w)) \tag{2.25}$$

where

$$Z_s(\alpha) = \int exp(-G - \alpha E_W) dw \tag{2.26}$$

Since the aim is to find the weights and the hyperparameters, the principle of marginalisation is used to integrate out all unknown parameters.

$$p(w|D) = \int p(w, \alpha|D) d\alpha = \int p(w|\alpha, D) p(\alpha|D) d\alpha \tag{2.27}$$

The evidence framework uses the Laplace approximation that $p(\alpha|D)$ is sharply peaked around α_{MP} [27].

$$p(w|D) \approx p(w|\alpha_{MP}, D) \tag{2.28}$$

This hyperparameter α_{MP} thus optimises the weight posterior probability, and can be fixed to when making predictions.

The second approximation in the evidence approach is used when integrating $p(w|\alpha, D)$. MacKay suggests a spherical Gaussian distribution around the mode of the posterior. This approximation is local around a particular w_{MP} based on a second order Taylor series expansion.

$$S(w) \approx S(W_{MP}) + \frac{1}{2}(w - w_{MP})^T A (w - w_{MP}) \tag{2.29}$$

where A is a hessian matrix of the error function S with respect to the weights.

In summary, the initial hyperparameter α is set to a small arbitrary value, to allow the network to find patterns in the data before regularisation takes place [28]. Training takes place using the usual optimisation algorithms with the aim of minimising the cost function. When the training error falls below a certain tolerance, the hyperparameter is re-estimated, and thus the parameter is evaluated on line. The network is incrementally trained, and the parameters are re-estimated, and the cycle continues until a minimum tolerance is reached.

In this study, a single hyperparameter is used, which corresponds to the standard global weight decay scheme [28]. Multiple hyperparameters are used when there is a need for different hyperparameters in the hidden and outer layer weights, or biases. This need arises when the number of inputs differs greatly from the number of hidden nodes.

Making Predictions

Unlike the distribution of α we cannot assume that the weight posterior is sharply peaked around w_{MP}, since the output function $y(a)$ is logistic and not linear. The prediction therefore cannot be assumed to be the most probable output. The output in this study is the "moderated output" defined by MacKay [29], and the derivation follows. MacKay assumes that a is locally a linear function of the weights [29], with a distribution given by 2.30 [27].

$$p(a|\tilde{x}, D) = \int p(a|\tilde{x}, \tilde{w})p(\tilde{w}|D)d\tilde{w} \tag{2.30}$$

Assuming the weight posterior to have a Gaussian distribution with mean a_{MP} and a variance of

$$s^2(x) = g^T A^{-1} g \tag{2.31}$$

where g is the gradient of a with respect to the weights at w_{MP}. Then using 2.30 the probability that an input vector x̃ belongs to class C_1 is:

$$P(C_1|\tilde{x}, D) = \int P(C_1)p(a|\tilde{x}, D)da = \int f(a)p(a|x, D)da. \tag{2.32}$$

which is analytically intractable, so using MacKay's approximation becomes

$$P(C_1\tilde{x}, D) \approx f(\kappa(s)a_{MP}) \tag{2.33}$$

where

$$\kappa(s) = \left(1 + \frac{\pi s^2}{8}\right)^{-1/2}. \tag{2.34}$$

Chapter 3

Optimisation Algorithms

3.1 Genetic Algorithm

3.1.1 Background

Problem solving systems based on the principles of evolution and hereditary have been developed over the past thirty years [30]. These systems consist of populations of potential solutions from which certain individuals are selected based on fitness of individuals, and the individuals are altered by genetic operators. Some examples of such systems include: Evolution Strategies, Evolutionary Programming, Scatter Search Techniques, Genetic Programming and Holland's Genetic Algorithms [30].

Genetic algorithms, first proposed by Holland in 1975, are stochastic global optimisation methods of searching for a solution, typically in large search spaces where classical techniques are insufficient [31]. They have proved to be powerful for multi-variable optimisation when it is not possible or difficult to write the analytical form of the error function in large multi-modal spaces. The process is based on Darwinian evolution, so the terminology is borrowed from natural genetics.

The population consists of *individuals or genotypes*, often called *strings or chromosomes.* We consider each individual to have only one chromosome (although this is not the case for many organisms), so one chromosome and thus one individual represents a complete solution. *Genes* are the units that make up the chromosome, each gene controlling a specific *feature, character*, and these features can have different possible states, called *alleles.* Chromosomes with the better characteristics survive, thus successive generations of chromosomes improve in quality, as long as the criteria for survival is appropriate [32].

When applied to optimisation, each individual represents a solution to the problem and the evolution process on a population of individuals constitutes the search through the space of solutions. Gradient based methods use the best solutions to determine movement direction, in contrast, random searches explore the search space ignoring the best solutions. The GA therefore combines the merits of both these methods: exploration of the search space while exploiting the best solution. GA's are thus more robust than directed search methods.

Another advantage GA's have over gradient methods is that they perform multi-directional searches, as they maintain a population of solutions. Gradient based methods iteratively improve a single point: a new point is chosen from the neighbourhood of the initial point and becomes the current point if the value of the objective function improves. If the value does not improve, a different point is chosen and tested. The success of the gradient based method therefore depends heavily on the initial point, and it finds local minima only. No information regarding the relative error with respect to the global optimum is available in the gradient methods. Additionally, information is formed and exchanged between the directions suggested by the individuals.

3.1.2 Definition

The evolution program is a probabilistic algorithm which tracks a population of individuals at iteration t $P(t) = x_1^t, \ldots, x_n^t$. Each solution is encoded as an individual, and is evaluated to give a measure of "'fitness"'. The next generation will contain the more fit individuals of the current generation (chromosomes of the current generation are *reproduced*), and some individuals will be transformed by the genetic operators, mutation and crossover, to form new solutions. The GA is thus fully described by the following components: representation, method of creating the initial population, evaluation function, genetic operators, and values for the parameters that the algorithm uses [30].

3.1.3 Representation

Chromosomes can be represented by real numbers, permutations of elements, a list of rules, or other symbols [32]. Float encoding is used in this study as it has been proven that the real valued GA is an order of magnitude more efficient in CPU time than the binary GA [30].

3.1.4 Initialisation

There are various initialisation methods, but the algorithm can be tested by moving from a randomly initialised population to a better adapted population, since the final solution will have been created through search and recombination rather than the initialisation procedure.

3.1.5 Evaluation Function

The optimisation algorithm operates on the fitness function, either maximising or minimising it. The "fitness" quantifies the the quality of the solution, and this is used to compare individuals. An important aspect is the normalisation of the variables and hence the result of the evaluation function. The genetic algorithm is sensitive to the normalisation technique, as if it stresses improvements too much, it could promote the dominance of a single gene. Conversely, the fitness of better individuals should clearly indicate the genetic superiority of these solutions, as compared to weaker individuals. A second feature of the GA is that the algorithm is independent of the the way in which the evaluation is performed, making it robust [31]. Compared to heuristic methods dependant on the domain and specific to the situation, the GA is flexible. Individuals of the population are selected for reproduction according to their fitness using a selection function.

3.1.6 Selection Function

Solutions with higher fitness are selected more frequently by the probabilistic selection function. There are different versions of the selection function: roulette, normalised geometric selection and tournament. Even though the roulette selection method is random, each parents chance of being selected is directly proportional to its fitness. Roulette selection sometimes emphasises superior chromosomes, and they could distort the gene pool, since the population is finite. This is undesirable, a leading cause of premature convergence. Rank-based selection, normalised geometric ranking in the study [30], tends to converge slowly with less premature convergence and better gene pool diversity. The tournament selection does not use probability, instead n individuals are selected, and the fittest individual of the tournament is selected. The process is repeated until the required number of individuals for the next generation is obtained.

3.1.7 Genetic Operators

When two individuals are selected for mating, two genetic operations can be performed on their chromosomes. *"Crossover"* mixes two chromosomes of the parents, by cutting the chromosomes at random points along their length, and exchanging the cut sections. This produces two new chromosomes, each having characteristics of the parents. Both new chromosomes are placed in the gene-pool and either replace poorer quality chromosomes, or are discarded. For the float-encoding of chromosomes there are 3 different crossover techniques: simple, arithmetic and heuristic. Simple crossover cuts and exchanges two sections of the chromosome at a single random point. Arithmetic crossover takes two parents X and Y and performs an interpolation along the line formed by the two parents to produce two complimentary linear combinations of the parents.

$$C_1 = rX + Y(1 - r) \tag{3.1}$$

$$C_2 = rY + X(1 - r) \tag{3.2}$$

Heuristic crossover creates two new individuals based on fitness. One child will resemble the fitter parent, and the other will be a combination of the parents, but will have more genetic material from the fitter parent. This is given by equations 3.3 and 3.4. Depending on the value of r, C_1 could be infeasible, so a new value of r can be generated, a maximum number of retries can be set, with C_1 set to X if all attempts fail.

$$C_1 = X + (1 - r)Y \tag{3.3}$$

$$C_2 = Y \tag{3.4}$$

"Mutation" alters one characteristic on the chromosome. It occurs on a very small portion of the population. Without mutation, eventually all the solutions would be exactly identical. Sometimes the mixing of chromosomes with different characteristics results in a better solution, and mutation enables this to happen. One variable is replaced with either: a uniform random number, a non-uniform random number, the lower or upper bound, or all variables of one parent can be replaced with non-uniform numbers. More detail can be found in [33]

3.1.8 Algorithm

Different solutions are encoded as individual chromosomes. and each different variable is represented by a gene. The process of reproduction advances the species by producing members of the successive generation. At first, completely random solutions are tried and evaluated according to a fitness function, and then the best ones are combined using specific operators. This gives the ability to adequately explore possible solutions while, at the same time, preserving from each solution the parts which work properly.

At each iteration t a genetic algorithm maintains a population of potential solutions (chromosomes), $P(t) = x_1^t, \ldots, x_n^t$. Each solution is evaluated using the objective function to obtain a measure of its fitness. Selection of individuals for the next iteration (generation) is based on the fitness, with preference given to the more fit individuals. New solutions are formed by altering the chromosomes through crossover and mutation.

The basic genetic algorithm is as follows:

1. Initialise population
2. Evaluate and select individuals for mating
3. Mate individuals to produce offspring
4. Mutate offspring
5. Evaluate offspring
6. Insert offspring into population
7. Evaluate stopping criteria, if satisfied return to 2 else proceed to 8
8. Finish

3.1.9 Parameters

- Encoding: binary or float representation of the chromosome
- Population size
- Maximum number of generations or stopping criteria
- Mutation: type and rate (number per generation)
- Crossover: type and rate (number per generation)
- Selection: method and probability of selecting the best

Table 3.1: Particle Swarm Optimisation analogues

Optimisation Concept	Particle Swarm	GA
Potential solution	Particle	Chromosome
Objective Function	Fitness	Fitness
Aim	Food location	Fittest individual
Control parameter	Velocity	

While the GA is an attractive option for global optimisation, in some cases the GA will be unable to find the optimal solution. Such instances are due to premature convergence. If convergence occurs too rapidly, potential solutions in part of the population are often lost [30] Although a more comprehensive search is possible with infinite generations and individuals in the population, this is limited by computation time and resources, and the limitation also affects accuracy of the solution.

3.2 Particle Swarm Optimisation

3.2.1 Background

Particle swarm optimisation (PSO), a population based stochastic optimisation technique, originated from the intent to graphically simulate a flock of birds, and the motive for this simulation was to model human social behaviour. The algorithm, first proposed by Kennedy and Elberhart, is related to genetic algorithms and evolutionary programming [34]. The first models represented the dynamics of a flock of birds in search of food, where the most effective strategy for the birds to find the food is to follow the ones nearest the food. To do this, the models initially only represented the velocities of the birds, and as they progressed, began to include memories of the best position of the individual as well as the group.

The hypothesis for the success of the flock is that individual members can benefit from the previous experience of all other members of the group during the search for food. This sharing of information provides the group with a greater collective advantage than the disadvantage per individual due to competition, and is an evolutionary advantage. After simplification of the concept, the flock terminology became replaced with a swarm. The particle swarm optimisation method is similar to the GA, and the optimisation analogues of the concepts are presented and compared in Table 3.1.

3.2.2 Algorithm

The swarm consists of particles that each represent a solution, and the positions of these particles are randomly initialised. The aim is for the swarm to fly in the search space, and accelerate toward the best solution, or food source. This is achieved by adjusting the velocity of the particles as they explore the search space, to follow the particle nearest the best solution. Each member of the swarm is initialised with a velocity. The information known at each iteration is the present position, the best position for that particle so far, and the best position attained by the group [34]. At each iteration, the position x of each particle is evaluated, and the velocity v is updated according to the known information, as shown in equations (3.5) and (3.6). The inertia weight w controls the impact of the previous history of velocities on the current velocity. Larger inertial weights facilitate in global exploration (searching new areas) while smaller inertial weights are used for local exploration. The optimal weight balances the two types of searches, reducing the time required to find the optimum.

$$\tilde{v}^i_{k+1} = w\tilde{v}^i_k + c_1 rand(p^i - \tilde{x}^i_k) + c_2 rand(\tilde{p}^g_k - \tilde{x}^i_k) \tag{3.5}$$

Where v is the velocity, k is the iteration, i describes the particle, w is the inertial weight, x is the position, and g is the index of the best particle of the swarm.

$$\tilde{x}^i_{k+1} = \tilde{x}^i_k + \tilde{v}^i_{k+1} \tag{3.6}$$

The pseudocode is as follows:

```
FOR each particle
        Initialise particle
END
DO
        FOR each particle
         Calculate fitness value
         If fitness value is better than best fitness value $p_{best}$
         in history
           set current value as the new $p_{best}$
         END
Choose particle with best fitness value of all particles as $g_{best}$
        FOR each particle
          Calculate particle velocity according to (3.5)
          Update particle position according to (3.6)
```

```
    END
WHILE maximum iterations or minimum
error criteria is not attained
```

Although crossover and do not exist in particle swarm optimisation, the concepts are employed through the acceleration of each particle toward its best position as well as the global best position. The biggest difference is that selection does not occur in PSO, all particles stay in the swarm for the entire duration of the optimisation [35]. Particle positions are influenced only by their own history, and the best position of the group. This is different from the GA, where individuals are directly influenced by random members (the children are affected by the parents)

3.2.3 Parameters

Design parameters in particle swarm optimisation are: the number of particles, the dimension of the particles (depends on the number of variables), the range of particles (range of the variables), the maximum velocity, learning conditions and the stop condition. The maximum velocity is the maximum change one particle can take during one iteration, this is usually set as the range of the particle.

Chapter 4

Classification as Statistical Pattern Recognition

The goal of classification is to assign an object to a predefined group or class, based on observed attributes related to the object [36]. The observed attributes arranged in a sequence constitute a pattern, thus the task of classification involves pattern recognition. The ability to classify different objects lies in the fact that there are enough differences in certain attributes to distinguish one class of object from another. These differences are used to create different mathematical models that represent the objects. The process of classification starts by taking the observed information, and selecting the model that fits the observed data best. The class that the models belongs to is the designated class [37].

The design of a classifier thus involves the development of the class representative models, and of the algorithm that will correctly assign a model to an individual, given the observed data. Each individual is represented by a feature vector x describing the characteristics of that individual, and is assigned to to one of k classes, $C_1, \dots C_k$. The data on which the model is based contains examples of individuals, as well as the classes to which those individuals belong. The output of the classification system is assigned to the variable y. The classification model is therefore required to map the inputs $x_1, \dots, x_d$ to the output y. A mathematical function describes this mapping, and since it cannot be explicitly determined, the data is used to determine the parameters. This can be written as follows:

$$y_k = f(x; w), \tag{4.1}$$

where w are the mapping weights.

Instead of viewing classification as a functional mapping task, we can consider each individual to be represented by a feature vector, and classification is the partitioning of the feature space into the regions representative of each class. Design of the classifier is determining where to place the decision boundary to correctly separate the different classes.

An important issue in statistical pattern classification, is generalisation. If the classifier is designed using many features and few training samples, the decision boundary becomes intricate and more fitted to the training data. While this will improve accuracy of classification of seen examples (training data), when presented with new unseen (test data), the classifier will be unable to predict the classification of the samples. Generalisation refers to the ability of the classifier to correctly model the underlying behaviour of the data so that it is able to classify unseen data. The trade off in the design is between accuracy on unseen patterns and the simplicity of the classifier.

4.1 Bayesian Decision Theory

The statistical approach to the problem of pattern classification taken in this study, uses Bayesian decision theory. It is based on the assumption that the decision problem is posed in probabilistic terms and that all relevant probability values are known [37].

What probabilities are required? At the very least, given the data, the probability of a particular sample belonging to class C_1 is known. This probability is referred to as the a priori probability, and reflects the prior knowledge of the likelihood of class $C1$ occurring. The sum of the probabilities of all classes sums to one, and the individual class prior probabilities are non-zero.

Additional information is gained when the actual sample to be classified is analysed. The measurements of the sample (features) will contain information, which can be used to adjust the a priori information to become the a posteriori information. Consider a measurement x, t is a continuous random variable with a probability distribution dependant on the class, $P(x|C)$. Bayes rule is used to calculate the a posteriori probability,

$$posterior = \frac{likelihood \times prior}{evidence} \tag{4.2}$$

To make a decision based on the information, a decision rule is followed. The rule is defined in such a way to minimise the cost of misclassification. This rule is implemented mathematically by taking the features and partitioning the feature space to regions where each region contains the vectors representing that particular class.

4.1.1 Allocation Principles, or Decision rules

The Bayes Decision rule attempts to minimise the probability of error and hence the cost of misclassification.

4.1.2 Discriminant Functions

One way of describing the method by which the feature space is divided, is by a set of discriminant functions. Each class is represented by one discriminant function, and the classifier assigns the vector input to the class corresponding to the highest discriminant. Two class classification is a special case, and only one discriminant function is required. Instead of calculating the discriminant for each class, and assigning the class to the highest discriminant, one discriminant function calculates the difference between the two discriminants.

$$g(x) = g_1(x) - g_2(x) \tag{4.3}$$

4.2 Bayesian Classification with Neural Networks

In this study, we have a dataset consisting of different patterns, vectors in x, with corresponding outputs, y. Artificial neural networks are used to obtain the functional mapping between x and y, and supervised learning is used to obtain the parameters. This process is the training of the networks, and is called supervised learning since the data contains examples of known classification. The purpose of the classification model is to design the decision surface to assign new inputs to one of many discrete classes [38]. Another type of neural network classifier, not investigated here, uses unsupervised training to perform clustering when the classes are unknown.

There are two stages in classification: in the first stage, the posterior probabilities are inferred from the data. In the second stage, these probabilities are used to make the classification decision, and there are several methods for this process. The simplest and most common is to minimise risk by maximising the posteriori probability. Another approach is to consider the cost of misclassification for each different class, these values are then placed in a loss matrix and multiplied by the posterior probabilities.

The neural network estimates or rather, models the class conditional densities, the probability distribution of the vector given the class information. Using Bayes theorem, the posterior probability for new unseen values of x can be used to classify x.

Classification can be either discriminative or generative. Discriminative classification methods model the conditional distribution of the class, given the inputs, while generative classification models the joint distribution of the classes, and prior class distribution directly. Neural network classification can perform either generative or discriminative classification. Generative classification uses discriminant functions. If the network is used to represent a non-linear discriminant function, when presented with the input vector, the network produces the classification directly [21]. Alternatively, the network can be used to model the posterior probabilities of the class membership.

Whether the generative or discriminative classification approach is chosen affects the functionality of the neural network and hence the design of it. When modelling the posterior probabilities, there must be one output per outcome, or class so that each activation represents the posterior probability of each class. In other words, each output represents $p(C_k|x)$, where C_k is the kth class and x is the input vector. The advantage of modelling individual posterior probability of class membership is that it allows us to compare the posterior probabilities for each outcome, allowing minimum error-rate decisions to be made.

4.2.1 Decision boundaries and the decision rule

The output of the neural network is the posterior probability, and this needs to be transformed to a classification decision rule. There are various methods of doing this, one of which is to accept the class with the highest probability, or the maximum a posteriori rule.

The rule for minimising misclassification can also be seen in terms of decision regions. The aim of classification is to assign the points in the feature space to each class.

The feature space can thus be divided into decision regions, with the boundaries separating the regions termed the decision boundaries. The design of a classifier aims to find the optimal placement of these boundaries to minimise misclassification.

4.2.2 Density Estimation

The neural network estimation of the probability density is a combination of the parametric and non-parametric methods, semi-parametric approach. Parametric methods assume specific functional forms of the density, and require optimisation of the parameters to fit the data. The problem with this method is that the chosen function may not be able to model the distribution accurately. Conversely, non-parametric methods model the density purely based on the data, and the problem is that the model may become too complex, with many parameters.

The semi-parametric method starts with a general function, and parameters controlling complexity of the model can be controlled independently of the dataset. Neural networks are an example of the semi-parametric method, and exhibit good scaling properties with larger datasets, this helps keep complexity of the model to a minimum, aiding in generalisation.

Chapter 5

Missing Data Estimation

Missing data is often a problem associated with survey and medical data [39]. Some reasons for omission are: data entry errors, non-response of participants and inapplicable responses. The reason missing data is such a problem is that statistical methods cannot perform where there is missing data. Depending on the extent to which it is missing, missing data can completely alter the nature of the data.

There are two ways of dealing with missing data, either deletion or imputation. Deletion is infeasible if there is a substantial amount of missing data. Imputation is the process of estimating missing data of an observation based on valid values of other variables. The current methods for imputation rely mainly on averages, or finding similar cases in the data to substitute with.

The ability to infer missing data from a particular dataset depends on the reason the data is missing. Different types of missing data were introduced by Rubin [39]: Missing at Random (MAR), Missing Completely at Random (MCAR), and non-ignorable. MAR data occurs when the distribution of missingness does not depend on the missing data, but only on the observed data. This also known as the Ignorable response, since this data may be predicted or found from data within the set. When the distribution does not depend on either the missing data nor the observed data, it is the special case of Missing at Random, Missing Completely at Random [40]. The Non-ignorable non-response is when data is missing not at random. In other words, the distribution of missing data is dependent on the missing data itself. The assumption in this study is that all the data is MCAR, since MAR data is non-ignorable and unpredictable.

MCAR data relies on the interrelationships between the known values. One way of discovering these interrelationships, is to use neural networks, which are able to derive relationships in complex data [21].

5.1 Method

The method used to approximate the missing clinical survey data is based on one method developed by Abdella and Marwala [19], in which an auto-associative neural network is trained to model the complete data. This type of MLP predicts the same inputs as the outputs. While this may seem redundant on a complete data set, when the input is incomplete, the auto-associative network is used to predict those missing entries. In the case of missing data, the input to the network consists of the partial set of inputs (held constant) and an initial guess of the missing inputs (variable). The neural network then predicts the output according to the dataset characteristics, hence the output of the trained network should approximate the input. The missing inputs are varied until the set of inputs closely resembles the set of outputs. In other words, the aim is to minimise the difference between the predicted values at the output of the network, and the applied input. This can be seen as Least Squares Constrained Optimisation, a minimisation of the least squares error function.

There are two main types of methods of optimisation: gradient methods, and stochastic methods. Both types of methods were investigated, and it was found that the most efficient gradient based methods are second order, with the secant method being the most efficient. However, the stochastic methods have a much shorter computation time and have higher probabilities of finding global solutions, and this report focuses on this type of optimisation. Since each entry represents an individual multivariable optimisation problem, total execution time is a critical performance measure, as on the survey data set used in the study 284 out of the 3381 cases were incomplete. Several stochastic methods were tested, namely genetic algorithm, particle swarm optimisation, and simulated annealing. Of these different methods, particle swarm optimisation executed and converged in the shortest time.

The system consists of two main parts: the first is the auto-associative neural network discussed in Section 5.2, and the second is the optimisation algorithm in Section 5.3. Thus, the performance of the system depends on two factors:

1. the ability of the neural network to accurately predict the output
2. the ability of the optimisation algorithm to find a global minimum

5.2 The Auto-associative neural network

An MLP with the same number of inputs and outputs as there are variables in the dataset is trained on a full dataset, where there are no entries missing. The weights and biases are hence fixed for the network to be able to predict itself.

The output (Y) of the neural network can be written as follows:

$$\widetilde{Y} = f(\widetilde{X}, \widetilde{W}) \tag{5.1}$$

Where $\widetilde{X}$ is the input vector, and $\widetilde{W}$ is the vector of weights.

The performance of the auto-associative neural network is dependant on the architecture and training. Since the maximum number of hidden nodes permissible is the number of inputs and outputs, it was possible to design the network by trial and error, as the upper limit on nodes is 13.

5.3 Least Squares Error Optimisation

5.3.1 Least Squares Error Function

The error function, or objective function is the sum of squared difference between each corresponding input and output.

$$E = \sum_{n=1}^{N} (\widetilde{X} - \widetilde{Y})^2 \tag{5.2}$$

Substituting Y from equation (5.1), the error is:

$$E = \sum_{n=1}^{N} (X - f(\widetilde{X}, \widetilde{W}))^2 \tag{5.3}$$

Since some data is missing, $\widetilde{X}$ can be split into the known elements $(\widetilde{X_k})$ and the unknown elements $(\widetilde{X_u})$. Rewriting equation (5.3) [19]:

$$E = \sum_{n=1}^{N} \left(\left\{ \begin{array}{c} \widetilde{X_k} \\ \widetilde{X_u} \end{array} \right\} - f\left(\left\{ \begin{array}{c} \widetilde{X_k} \\ \widetilde{X_u} \end{array} \right\}, \widetilde{W} \right) \right)^2 \tag{5.4}$$

5.3.2 Gradient Optimisation

Gradient of the error function

The gradient is calculated for the missing variables only, that is, the derivative of the error function with respect to the missing variables only. Although the input and output vectors can be categorised as *known* and *unknown* terms, for simplicity the gradient of the error function is defined with the entire $\widetilde{X}$ and similarly for $\widetilde{Y}$. For the i^{th} input, the gradient of the error is shown in equation(5.5).

$$\frac{\partial E}{\partial x_i} = \sum_{n=1}^{N} 2\left(\widetilde{X} - f(\widetilde{X}, \widetilde{W})\right)\left(\frac{d}{dx_i}\widetilde{X} - \left(\frac{\partial y_1}{\partial x_i} + \ldots + \frac{\partial y_n}{\partial x_i}\right)\right) \tag{5.5}$$

Since $\widetilde{X}$ is a vector of constants, the derivative of $\widetilde{X}$ with respect to any of the missing variables will be 1, for that specific component in $\widetilde{X}$. Thus, the gradient is simply:

$$\frac{\partial E}{\partial x_i} = \sum_{n=1}^{N} 2\left(\widetilde{X} - f(\widetilde{X}, \widetilde{W})\right)\left(1 - \left(\frac{\partial y_1}{\partial x_i} + \ldots + \frac{\partial y_n}{\partial x_i}\right)\right) \tag{5.6}$$

To obtain the partial derivatives of y the Jacobian of the system function must be calculated. In this case y refers to the output vector $\widetilde{Y}$, so the derivative of each component of $\widetilde{Y}$ with respect to each missing variable must be solved for. The calculation of the neural network Jacobian is done by backpropagation [27].

The Jacobian is defined by:

$$J_{ki} = \frac{dy_k}{dx_i} \tag{5.7}$$

Starting at the output, from equation(2.4), defining the output at a linear activation function, the derivative of y_k with respect to the input x_i is:

$$\frac{dy_k}{dx_i} = \frac{da_k^{(2)}}{dx_i} \tag{5.8}$$

The output activation of the second layer is expressed as a function of the transformed activations from the hidden layer, z_k. From equation(2.3) $a_k^{(2)} = f_1(z_k)$. These transformed activations in the hidden layer are a function of the activations from the input layer to the hidden layer $a^{(1)}$. From equation (2.2), $z_k = f_2(a^{(1)})$. Similarly, moving further backward, the first layer activations are a function of the inputs x_i, $a_i^{(1)} = f_3(x_i)$.

The Chain rule is used:

$$\frac{da_k^{(2)}}{dx_i} = \frac{da_k^{(2)}}{dz}\frac{dz}{da_j^{(1)}}\frac{da_j^{(1)}}{dx_i} \tag{5.9}$$

Differentiating the activations in the second layer, equation (2.3), with respect to the outputs z from the hidden layer gives:

$$\frac{da_k^{(2)}}{dz} = \sum_{j=1}^{h} w_{kj}^{(2)} \tag{5.10}$$

Differentiating the output hidden activations, equation (2.2), with respect to the activations at the hidden layer $a^{(1)}$ gives:

$$\frac{dz}{da_j^{(1)}} = sech^2(a_j^{(1)}) \tag{5.11}$$

Using the identity $sech(a)^2 = 1 - \tanh(a)^2$, equation (5.11) becomes:

$$\frac{dz}{da_j^{(1)}} = (1 - \tanh(a_j^{(1)})^2) \tag{5.12}$$

The first layer activations differentiated with respect to the input x_i

$$\frac{da_j^{(1)}}{dx} = \sum_{i=1}^{n} w_{ji}^{(1)} \tag{5.13}$$

The final equation for the Jacobian is thus as follows:

$$\frac{dy_k}{dx_i} = \sum_{j=1}^{h} w_{kj}^{(2)}(1 - \tanh^2(a_j^{(1)}))\sum_{i=1}^{n} w_{ji}^{(1)} \tag{5.14}$$

Where: $a_j^{(1)}$ is given by equation(2.1).

The Jacobian for the neural network is substituted into equation(5.6). For a problem with m missing variables, the gradient will be a vector of length m. Only the derivatives of y with respect to the missing variables will be calculated.

To approximate the gradient, each component of the derivative is obtained by perturbing one missing variable by a finite difference.

Optimisation Methods

General purpose optimisation algorithms in the Netlab toolbox were used: scaled conjugate gradient, conjugate gradient, quasi-Newton, and gradient descent. The secant method is derived from the quasi-Newton method.

More detail on these gradient based optimisation algorithms is supplied in Appendix C, with the performance testing on the implementation of the methods for missing data approximation in Appendix D.

5.3.3 Stochastic Optimisation

Methods

The different stochastic optimisation algorithms that were investigated are: particle swarm optimisation, genetic algorithm, and simulated annealing. The theory behind particle swarm optimisation and genetic algorithms is in Section 3. Simulated annealing was investigated but was not as successful as the particle swarm optimisation method, and the theory is found in Appendix E. The results from the investigation into various stochastic optimisation methods for multivariable optimisation in missing data approximation is in Appendix F.

Simulated Annealing

Consistent results were obtained for this method, but it is computationally intensive, and the execution time is greater than all the other methods. The major disadvantage of simulated annealing is that the optimisation is performed on one solution. In contrast, the GA and PSO optimise many solutions, and this speeds up the optimisation. To counteract the possibility of an incorrect initial temperature, several annealing runs were carried out. However, these are performed serially, hence the execution time is much longer than that of the other methods. In addition, each annealing run has a longer convergence time, due to the slow cooling schedule.

Genetic Algorithm

The mutation rate had a more pronounced effect on the accuracy than the crossover rate or population size. The precision of the solution is not as important as evaluating much of the search space, as due to the rounding of the solutions, precision is lost anyway.

Particle Swarm

The PSO success can be attributed to the group interaction and search space is effectively broken down into areas covered by each agent. Since the algorithms produce similar results in terms of accuracy, the most efficient algorithm is the particle swarm optimiser, with an execution time of 35% of the time required by the GA. The second part of the testing is performed on unseen testing data, with varying number of variables missing.

Chapter 6

Implementation

6.1 Neural Network Classifier

The steps involved in designing a classifier are:

1. Data Collection and Processing: The survey data used was collated and processed to make it suitable for neural network training

2. Feature Choice: Since there were few distinct features present in the data, all features were used. This could be done because although the neural networks do not eliminate unecessary features altogether, they adjust to the data by assigning larger weights to more relevant features.

3. Model Choice: This step involves the design of the neural network architecture. Several different architectures (MLP and RBF) and training methods (Maximum likelihood training, and variants of Bayesian Regularisation) were investigated in this study. A GA was used to select the parameters for training and the architecture of an MLP.

4. Training Evaluation: Performance of the classifier depends on its ability to model the data correctly with the ability to generalise. Assessment of classification accuracy was made with the Receiver Operating Characteristic.

The data was provided by the South African Department of Health in spreadsheet format, and was processed in MATLAB. The neural networks were also simulated in MATLAB, using the Netlab Toolbox.

6.1.1 Data Processing

Data Source

Demographic and medical data came from the South African antenatal seroprevalence survey of 2001. This is a national survey, and any pregnant women attending selected public health care clinics participating for the first time in the survey were eligible to participate. Anonymity is guaranteed. The antenatal seroprevalence surveys are used as the main source of HIV prevalence data worldwide, reasons for this are that antenatal clinics are found throughout the world, and pregnant women are ideal candidates for the study as they are sexually active. Antenatal refers to the pregnant women and seroprevalence is the level of a pathogen in a population, measured in blood serum. Information was obtained using a questionnaire, and the HIV status of the patient was measured using a enzyme linked immunosorbent assay (ELISA) test.

Missing Data

Out of a total 3381 cases, 3097 complete cases were selected, and the incomplete entries were discarded. An alternative to omitting data is to approximate it, and this system is discussed in Section 6.2.

Variables

The variables obtained in the study are: race, region, age of the mother, age of the father, education level of the mother, gravidity, parity and HIV status [41]. The qualitative variables such as race and region are converted to binary values in order to prevent placing an incorrect importance on these variables had they been coded numerically. The age of mother and father are represented in years. The integer value representing education level represents the highest grade successfully completed, with 13 representing tertiary education. Gravidity is the number of pregnancies, complete or incomplete, experienced by a female, and this variable is represented by an integer between 0 and 12. Parity is the number of times the individual has given birth, (for example, multiple births are counted as one) and this is not the same as gravidity. Both these quantities are important, as they show the reproductive activity as well as the reproductive health state of the women. The HIV status is binary coded, a 1 represents positive status, while a 0 represents

Table 6.1: Summary of Input and Output Variables

Variable	Type	Range
Input variables		
Region: A	Binary	0-1
Region: B	Binary	0-1
Region: C	Binary	0-1
Age	Integer	15-49
Race: African	Binary	0-1
Race: Coloured	Binary	0-1
Race: White	Binary	0-1
Race: Asian	Binary	0-1
Education	Integer	0-13
Gravidity	Integer	0-12
Parity	Integer	0-12
Age of father	Integer	15-54
Output variable		
HIV status	Binary	0-1

negative status. Thus the final number of input variables is 12, shown in Table 6.1. There is one output.

Outliers

Age is the only variable with outliers. The standard age bracket used in demographic studies relating to female fertility is 15-49 in African countries, and this was used to extract outliers in mother's age. The mean difference in age between mother and father is 5 years, and the upper limit on age of the father is thus 54. The data is partitioned into 3 sets, training, validation and testing.

Dataset biasing

The training set is balanced to consist of an equal number of positive outcomes as negatives, by duplicating the positive entries. An alternative to oversampling the minority class is to assign distinct costs to training examples, or by undersampling the majority class [42]. Due to the limited size of the dataset, over sampling the positive cases was used rather than undersampling the negative cases. The original training set consisted of 309 positives and 723 negatives. Neural networks are trained to model the statistical properties of the data, and had the neural network been trained on this biased dataset, the predicted outcome would always have been negative. This data was randomised and the inputs were scaled between -1 and 1. There are 1416 entries in this set. The validation and testing sets each contain 1027 entries.

6.1.2 Model Selection

Details of the investigation into different design and training methods may be found in Appendix A. The results show that the MLP is a more successful classifier than the RBF. The classification ability for the maximum likelihood trained MLP is marginally greater than other methods, but these findings were inconclusive, as the combinations of design parameters was insufficiently, covered to varying degrees for the various methods. Since the Bayesian method of training provides a structured approach to solving for weights and hyperparameters, this method was further investigated, using a Genetic Algorithm to automate the selection of the training and architecture parameters, as discussed in Section 6.1.2. The evidence procedure was selected over the Markov Chain Monte Carlo (MCMC) and Hybrid Markov Chain Monte Carlo (HMC) methods, since it is an approximation to the marginalisation procedure, reducing computation time and expense. Details on the theory behind MCMC and HMC is provided in Appendix B.

The Evidence procedure was applied using Netlab [27]. The number of weight optimisations is controlled by a the number of outer loops, and each weight optimisation consists of a number of evidence re-estimations controlled by the number of inner loops. The weights are optimised with a Bayesian error function, and the hyperparameters are re-estimated using the evidence procedure. The network produced contains weights set to a minimum in the error surface, and the hyperparameters α and γ are set to optimal values.

The algorithm of the evidence procedure is as follows [27]:

1. Initialise the hyperparameter α and the weights.
2. Train the network using scaled conjugate gradient optimisation method to minimise $S(w)$ of the Bayesian error function in Equation 2.29.
3. When a local minimum is found on the Bayesian error surface, evidence for the hyperparameter α is computed, and iteratively re-estimated using the Gaussian approximation.

Genetic Algorithm

A *Genetic Optimisation* Toolbox [33] was used for the implementation of the GA. The encoding of the algorithm is that each individual of the population represents the design and parameters of the evidence estimation procedure, of a Bayesian neural network. Genes represent the number of hidden nodes, the number of outer loops, and the number of inner loops. Two approaches to the fitness function were used: minimisation of prediction error (MSE), and maximisation of the area under the ROC curve.

The algorithm maximises the function, so the objective function is multiplied by factor of -1 to perform minimisation. Selection of the number of generations and population size was done to maximise exploration of the search space, while simultaneously maximising accuracy. Crossover produces new solutions, and together with selection ensures that more accurate solutions are found. Mutation is required to ensure the population does not converge prematurely. There is a trade-off between the evolution and the time taken to perform the calculations, more generations allows for more mutation and crossover, but this takes time. The parameters are adjusted to minimise the error without the population converging prematurely. Although limited crossover and mutation will ensure a slow and steady decrease of the error, there is a trade off in the time taken for a slower evolutions to execute.

The parameters used are:

- Selection: Normal Geometric
- Population size: 20 individuals
- Maximum number of generations: 100
- Crossover: 12 individuals per generation using arithmetic crossover

- Mutation: 16 individuals per generation using all methods of mutation, (refer to Section 3.1 of Chapter 3)

6.1.3 The ROC Curve

There are two methods for plotting the ROC curve, in the simple method, different thresholds are used to calculate the true positive and false positive ratios. The limitation is that since only a finite number of thresholds can realistically be calculated, points are omitted. The second method ranks test instances according to the score (probability) in descending order. Starting at (0,0), if the ranking is positive moves a step $1/pos$ up, else if the ranking is negative, moves to the right by an amount $1/neg$. The second method is implemented in this study, since the local concavities are smoothed by the first method of plotting, and these concavities indicate performance as explained in the previous section.

There is a trade off between sensitivity and specificity, and in this study equal importance was given to both properties. Given classification costs for either class, the optimal point could have been chosen to minimise a specific cost. The optimal operating point is on the convex hull of the curve, and the optimal accuracy is calculated at this threshold value. The minimum criteria for the selection of the operating point is that the true positive rate is greater than 0.5 and false positive rate is less than 0.5. The area is calculated using trapezoidal integration.

6.2 Missing Data Estimation

The numerical data values were estimated using the particle swarm optimisation algorithm, but for the categorical variables such as race and region this was not necessary. Instead, each option was tested, and the one resulting in the lowest error was selected as the best aproximation.

6.2.1 Constraints and scaling

The data used to train the MLP optimisation is scaled between 0 and 1 to ensure that the weights are of order unity. The importance of an input is proportional to the size, since the activations are a sum of the inputs. It is thus important that all data have similar ranges.

Scaling is also important in optimisation. If variables are of different orders of magnitude, the problem may be insensitive to variations in one or more of the variables, and the objective function contours will be distorted due to poor scaling. Any optimisation algorithm will struggle to converge to the true solution in this situation. The reason distorted contours poses such a problem, is that this causes difficulties in selecting the step length for the numerical gradients [43]. This is particularly evident in the gradient descent method.

Since the data is scaled between 0 and 1, the constraints are simply: $x_i < 0$ and $x_i > 0$.

6.2.2 Objective function

The objective function is the mean square error of the potential solution. The parameters to be optimised are the missing data entries. Since the number of missing entries varies from case to case, the number of parameters varies as the number of missing variables varies.

6.2.3 Method

Data Processing

The data for 2001 is filtered to remove partial entries, and outliers are removed to aid in neural network training. The two age variables were used as the basis for entry removal according to a 3 sigma standard deviation. This corresponds to the standard for demographic studies relating to female fertility where the age group is 15-49, and the age group for men was taken to be 18-51.

The full set is partitioned into training, validation and testing data. Originally, the training set was balanced to have an equal number of HIV positive outcomes and negative outcomes. This distorted the dataset to contain many more entries where the ages were nearer to the mean. The result was that the network was unable to correctly predict ages in the brackets (15-20) and (35+). The final training set thus represents the distribution of outcomes of the entire data set.

Neural Network

Correlation co-efficients were found for each variable in relation to the others. The variables each depend on different sets, but when combined the total dependence was on all the variables. As a result, when optimising several missing variables, it is not possible to remove any inputs. The auto-associative network thus takes all 13 variables as input with 12 hidden hyperbolic tan nodes and linear output nodes was found to produce the lowest training and validation error. Training was performed by the scaled conjugate gradient method for 1000 cycles.

6.2.4 Optimisation Algorithm

There are two parameters in the *Particle Swarm Optimisation*: the swarm size and the number of iterations. The members of the initial swarm are random particles within the bounds of the parameter search space. A swarm of 15 particles adequately covered the search space and converged to an optimum in under 300 iterations. The algorithm could be improved by improving local neighbourhood information, as the swarm diverged from the optimum initially, and reduced the error function only slightly.

Chapter 7

Results and Discussion

7.1 Classification

The best architecture selected by the GA is an MLP with 26 hidden nodes, trained for 100 training cycles using the scaled conjugate gradient optimisation technique. The evidence is re-estimated 3 times after each set of 100 training cycles to improve the estimation of the hyperparameters. This re-estimation process was performed 5 times. The fitness of the network is based on the area under the ROC as well as the classification accuracy. The average fitness of the population of 20 individuals converges at the highest fitness value after 30 generations, as shown in Figure 7.1.

To evaluate the accuracy of this prediction, an ROC curve was derived. If the network is to be used directly for classification, then the threshold for creating a binary decision from the probability can be adjusted, as discussed in the following subsection.

7.1.1 Threshold adjustment

The neural networks are trained to predict a binary outcome, 1 for HIV positive and 0 for HIV negative. The result from the neural network, however, is a continuous value between 0 and 1 and this needed to be hard-limited to either 0 or 1. This was achieved by rounding the output to 1 if greater than a threshold and rounding it to 0 otherwise. The confusion matrix was calculated initially for a threshold of 0.5 on the training data, and this value was adjusted until the ratio of false positives to false negatives was approximately 1. Since the data set consisted of equal positive and negative outcomes, the classifier should produce equal false positive

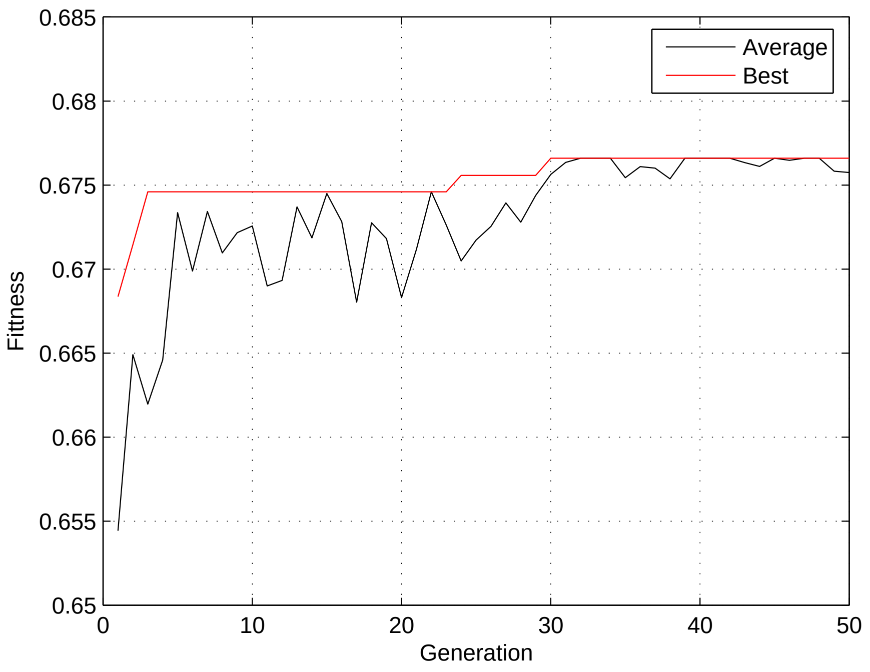

Figure 7.1: Improvement of Area Under ROC as fitness using GA

and false negative results, such that it is not biased toward predicting either case, in other words, specificity and sensitivity have equal importance. The networks were optimised for a unity false positive to false negative ratio. Using this adjusted threshold, validation and final testing are limited at 0 or 1. It is incorrect to adjust the threshold on the validation set, since this set is not balanced.

The best performing MLP network has a training AUC of 0.7385 and a validation AUCof 0.6701, as shown in Figure 7.2.

With an initial threshold of 0.5, the accuracy is 68% on training, but the number of false negatives (277) exceeds the number of false positives (169). The number of true positives is 539 and there are 431 true negatives. Shifting the threshold to 0.53 results in a better balanced classifier, with 242 false negatives and 210 false positives. The best performing RBF network has a validation ROC area of 0.6319. The initial threshold gives 276 false negatives and 246 false positives, so the threshold is shifted to 0.51, resulting in 257 false negatives and 266 false positives ontraining. There are 22 hidden nodes and the activation function is thin plate spline.

$$SimpleAcc = \frac{TP + TN}{TP + FN + FP + TN} \tag{7.1}$$

and

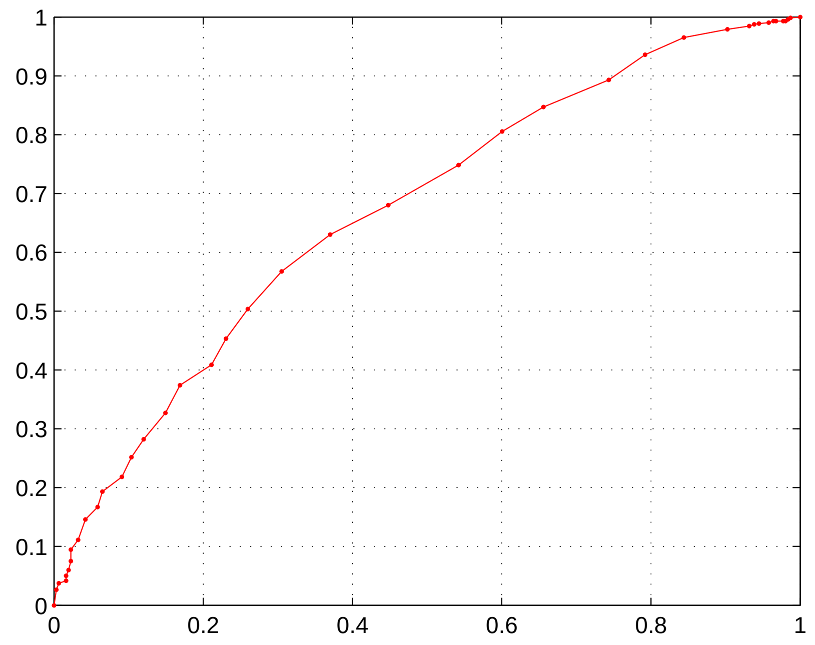

Figure 7.2: ROC for validation data using MLP

$$GeometricAcc = \sqrt{\frac{TP \times TN}{(TP + FN) \times (FP + TN)}} \tag{7.2}$$

7.2 Missing Data Estimation

Three aspects were tested:

1. The ability to predict different variables
2. The ability to estimate the data missing from the original dataset, (ability to predict combinations of missing variables).
3. The effect of the number of missing variables on the ability to predict missing data

7.2.1 Prediction of single missing variables

On a set of 200 complete entries, each variable was omitted and estimated using the particle swarm optimisation algorithm. The accuracy of prediction is given in Table 7.1

Table 7.1: Prediction of individual variables

Missing Variable	mse	max	min	rmse
Father's Age	25.285	12	19	4.005
Education	12.675	10	9	2.735
Mother's Age	20.95	12	17	3.53
Gravidity	0.685	6	-2	0.385
Parity	0.47	2	-1	0.33

The predictions are in an acceptable range of error. Although the maximum and minimum errors appear to be quite high, these occur for outliers, and this is illustrated in the histogram of errors.

7.2.2 Prediction of multiple variables

Tests were performed on cases with 1,2,3 and 4 missing values. Since the most efficient algorithm was found to be the secant method, this method was used for optimisation. The test data consists of 10 entries, and the data set is completely separate from the training data (as used previously). Different numbers of variables were removed each time.

The optimisation algorithm was run 5 times with different initial values. Different initial values had little effect on the solution in the first test, proving it to be robust. Discrepancies arise in the rounding of the solution, as all data are integer values. The best estimation of variable 12 are presented in Table 7.2.

The predictive ability is illustrated in Figure 7.3

The largest error is 15 years. The largest errors occur for the higher values, and it is suspected this is because there are fewer samples with the higher age values. More samples are needed to verify this.

$$error = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \tilde{x_i})^2}{n}} \tag{7.3}$$

This error is shown in Table 7.3 for the different number of missing variables. The error increases significantly when the third and fourth variables are removed. This increase is attributed to the fact that the third missing variable, the mother's age,

Table 7.2: Effect of number of variables missing per entry

	No. missing variables			
Data	1	2	3	4
41	52	51	54	56
21	23	24	24	21
61	50	48	54	56
40	35	32	33	34
20	22	22	24	21
32	31	31	27	25
36	36	37	34	36
30	30	30	31	31
20	22	22	19	16
30	25	25	26	25

is strongly correlated to the missing variable, the father's age. When the fourth variable is removed, the error decreases. It is expected to either equal the error for 3 missing variables, or increase, since less information is given to the neural network. This also means that there are more variables to optimise.

It is thus insufficient to test the performance on the number of missing variables alone. The combinations of missing variables is important, so correlation of variables must be analysed.

Table 7.3: Effect of number of missing variables

No. missing variables	1	2	3	4
error	3.2	3	4.3	3.9

7.2.3 Prediction of a combination of different variables

As the aim of the experiment is to approximate actual missing data, the simulated missing data represents the proportion of variables missing from the actual initial dataset. Cases of a single missing variable occurs most often for variable 12, the age of the father (153 cases out of 3381), then education level (53 cases out of 3381) variable 9. Cases of multiple variables missing occur most often for a combination of variables 4,9,10,12 (80 cases). There are therefore 4,53% entries with just the

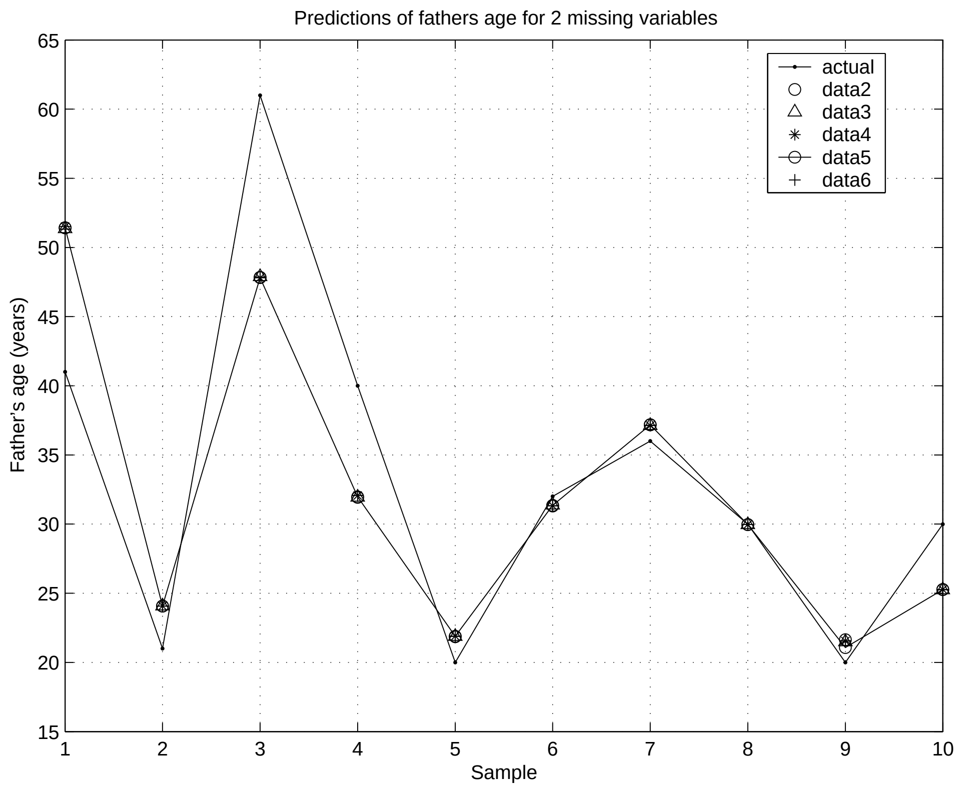

Figure 7.3: Predictive ability for variable 12 (Father's age)

father's age missing, 1.57% of the entries with education missing, and 2,37% with multiple variables missing. On the test set of 1027 entries, the simulated quantities are: 47 entries with age of father missing, 16 with education missing, and 24 entries with multiple missing variables.

Fifty entries of the testing set were approximated using the PSO optimiser, and the results are summarised in Table 7.4. The missing variables are Mother's age (years), Education level (grade), Gravidity (number), and Father's age (years). RMS error is calculated using Equation (7.3). The lowest error is 0 for all entries, but both the maximum and minimum errors are shown to illustrate the range of errors.

The error on predicted age of the father is too high to be used for actual imputation of health data, since the age brackets in fertility related studies are 5 years. However, since the ages of the mother and father are strongly correlated, it may not be possible to infer the age of the father without the age of the mother. In the case of 2 missing variables where the age of the mother is present, the accuracy of the prediction of the age of the father is acceptable. The age of the mother is accurate to within 5 years for all numbers of missing variables tested. The education level inaccuracy could be acceptable, but a distinction should be made between primary, secondary and tertiary education, since a 3 grade inaccuracy within any of these education

Table 7.4: The effect of missing data for approximation using the PSO algorithm

No. missing	Error	Mother	Education	Gravidity	Father
4	RMS	4.6562	3.1432	0.6782	5.8897
	Max	12	6	1	14
	Min	-9	-12	-2	-11
3	RMS	4.1905	2.884		5.8292
	Max	8	10		14
	Min	-8	-9		-12
2	RMS		2.7928		4.8990
	max		5		14
	min		-12		-11
1	RMS				4.8642
	max				12
	min				-11

levels has a significantly lower effect than if the error is across education levels.

7.3 Effect of Estimated Data on Classification

7.3.1 Single Missing Variable

Each variable from the complete set of 200 entries was omitted and approximated, and this data was used to predict the probabilistic outcome. The area under the ROC curve for the same complete set of 200 entries is 0.6004. The probabilities follow those produced using complete data, and this is shown in Figure 7.4.

7.3.2 Combination of Missing Variables

In the original data set, the most frequently occurring missing variable is the Father's age. The different proportions of the different variables was observed in the actual missing data, and replicated in the simulated missing data, as discussed in Section 7.2.3. This simulated missing data was used in classification and is compared to the

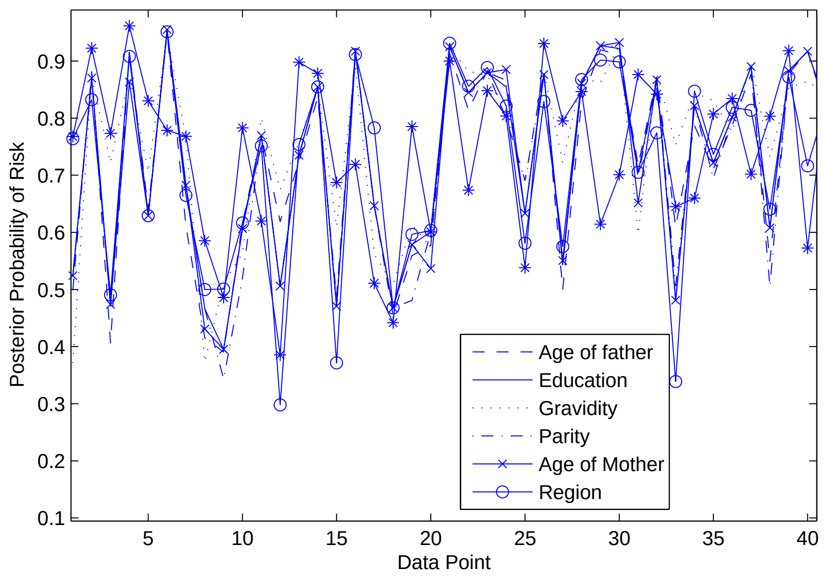

Figure 7.4: Predicted probability for approximated data (single variable)

Table 7.5: Effect of approximation of data on classification

Missing Variable	AUC
Father's Age	0.5895
Education	0.5916
Gravidity	0.5974
Mother's Age	0.6000
Parity	0.6075
Region	0.5377

results obtained from the same complete data.

Substituting the approximated data into the complete dataset, with 90 entries containing missing data and the remainder containing complete entries results in an AUC of 0.6520 while the AUC on the same set with complete data is 0.6566. The difference in area is marginal, and the ROC of the classifier, seen in Figure 7.6 using approximated data is very similar to that obtained with complete data in Figure 7.2.

The ROC curves were also generated for the data set consisting of the 90 entries of approximated data only. The performance of the classifier using approximated data differs significantly, with an AUC of 0.5427, compared to the AUC on complete data of 0.5885. For the threshold of 0.525, accuracy is 40% with 29 True Positives,

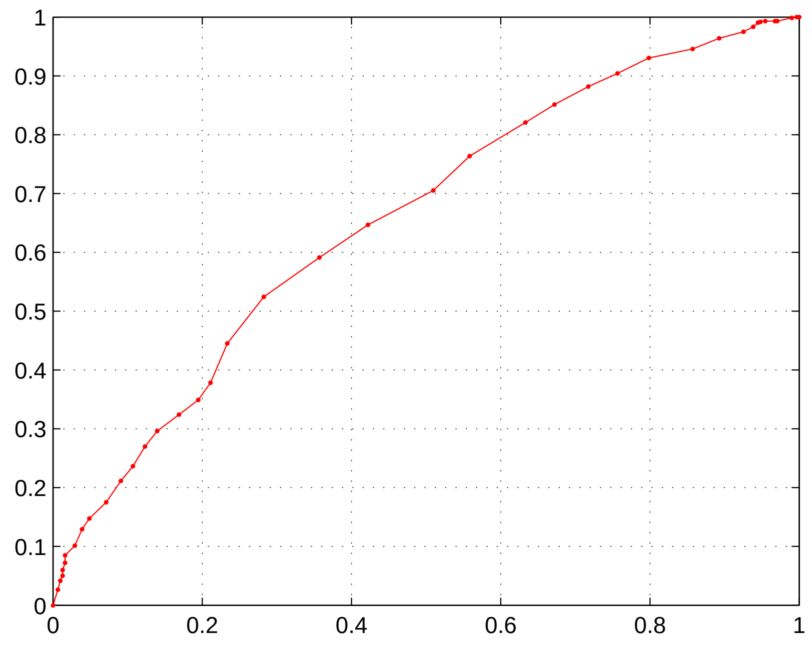

Figure 7.5: ROC for classifier using complete dataset

0 False Positives, 20 True Negatives and 21 False Negatives. The complete data on the same threshold gives an accuracy of 56.667% with 11 True Positives, 18 False Positives, 40 True Negatives, and 21 False Negatives. The predicted probability on approximated data is always higher than the threshold, resulting in the classifier predicting that each entry is positive.

The area under the curve gives an idea of the classification ability based on the classification of the whole data set. The predicted probability using the approximated data differs greatly from the predicted probabilities on the corresponding complete entries, with an MSE of 0.1489. Figure 7.7 shows that the predicted probability using approximated data is much higher than the prediction on complete data. The average probability is 0.7398, while on the complete data it is 0.4255.

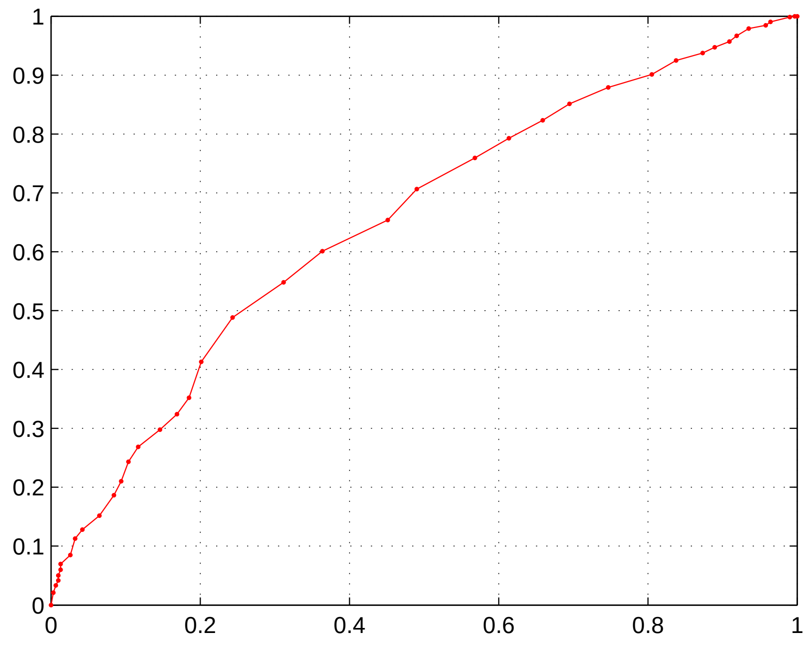

Figure 7.6: ROC for classifier using approximated dataset

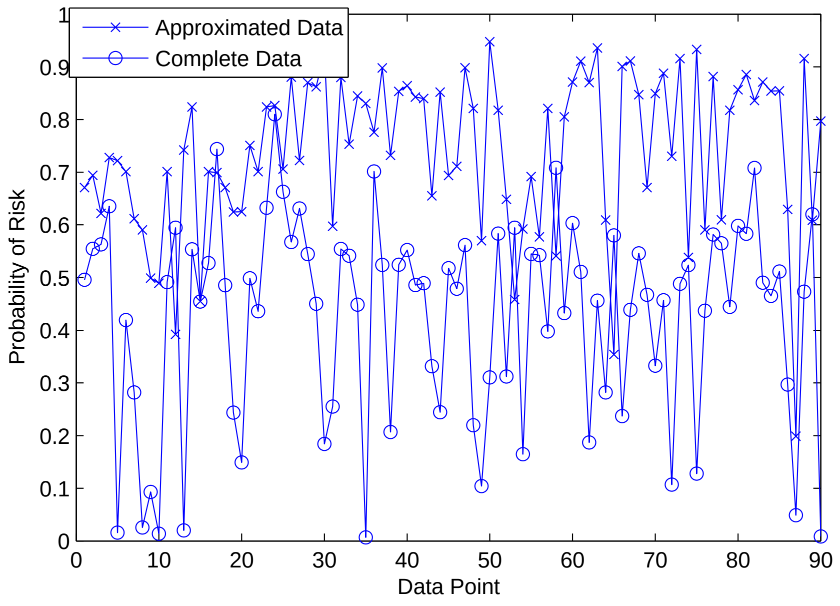

Figure 7.7: Predicted probability for approximated data (multiple variables)

Chapter 8

Conclusion

Artificial intelligence methods can be used for classification. In this study, supervised learning was used to train multilayer perceptrons and radial basis function networks to classify the HIV status of an individual, given certain demographic factors. Various aspects of data processing such as scaling and variable transformation are discussed. Using the maximum likelihood method of training, the problems encountered are that different weights are obtained from different training sets, and the weights are also dependant on the order of the data entries in the training set. To overcome this, a validation set of data is required to optimise the design of the network. This involves setting the regularisation parameters and the complexity of the network, set by the number of hidden nodes. The best network is chosen according to the lowest standard error between targets and predicted outputs, but the design process is time consuming since there are many permutations of parameters.

The uncertainty in the weights is captured by a probability distribution in the Bayesian framework. Hyperparameters representing the regularisation constants are determined empirically, and thus the Bayesian method is a more efficient design process. The number of hidden nodes, training cycles, and initial values for the hyperparameters must still be determined for this method, and these values are determined by trial and error. For the neural networks, maximum likelihood and Bayesian methods were used for training, and the theoretical basis for these methods is discussed in this report. Bayesian theory provides confidence intervals of the predictions, which in the case of classification manifests as the moderated or marginalised output. Performance metrics such as the area under the ROC curve, and both simple and geometric accuracy are used. Design of classifiers involves setting a threshold value to convert a probabilistic output to an actual classification. This value was determined by observing the ROC curve, and in this study, was set such that equal importance was given to sensitivity and specificity.

The results show that all neural network architectures produce similar results, but the neural networks trained with the Bayesian technique have marginally better accuracy and larger areas under the ROC curve. The average accuracy is between 61 and 62%, proving that demographic data is not sufficient to accurately predict HIV status, and this value is inadequate for medical classification. It is recommended that different input features be tested, as well as automatic relevance detection to assess which inputs contribute to the output. By observing these features, it would be easier to find additional relevant input features.

Estimation of the missing data did not affect the probabilistic output of the neural network classifier for single variable estimation, but was unsuccessful for predictions with multiple variable estimation. The missing data estimation system is thus recommended for cases where there is a single variable missing, but not in cases where there are multiple variables omitted.

References

[1] "UNAIDS Questions and Answers, Q&A II: Basic facts about the HIV/AIDS epidemic and its impact." URL `http://www.unaids.org/en/resources.asp`.

[2] S. Mitra, S. K. Pal, and P. Mitra. "Data Mining in Soft Computing Framework: A Survey." *IEEE Transactions on Neural Networks*, vol. 13, no. 1, pp. 3–14, 2002.

[3] D. Hand, H. Mannila, and P. Smith. *Principles of Data Mining*. Cambridge, Massachusetts: The MIT Press, 2001.

[4] O. Shisana. *Nelson Mandela HSRC study of HIV/AIDS: Full Report. South African national HIV prevalence, behavioural risks and mass media. Household Survey 2002*. South Africa: Human Sciences Research Council (HSRC), 2002.

[5] B. W. Zaba. "Adjusting ante-natal clinic data for improved estimates of HIV prevalence among women in sub-Saharan Africa." *AIDS*, vol. 14, pp. 2741–2750, 2000.

[6] S. Berry. "Understanding HIV and AIDS statistics.", May 2005. URL `www.avert.org`.

[7] R. Dorrington, D. Bradshaw, and D. Budlender. *HIV/AIDS profile of the provinces of South Africa indicators for 2004*. Centre for Actuarial Research, Medical Research Council and the Actuarial Society of South Africa, 2004. URL `http://www.commerce.uct.ac.za/care/`.

[8] A. Ntsaluba. "Summary Report: National HIV and Syphilis Antenatal sero-prevalence survey in South Africa." Tech. rep., Directorate: Health Systems Research, Research Coordination and Epidemiology 2002, Department of Health, 2002. URL `http://www.doh.gov.za/docs/reports/2000/hivreport.html#intro%`.

[9] L. Makubalo, P. Netshidzivhani, L. Mahlasela, and R. du Plessis. "Summary Report: National HIV and Syphilis Antenatal sero-prevalence survey in South Africa." Tech. rep., Directorate: Health Systems Research, Research Coordination and Epidemiology Department of Health, 2003.

[10] D. A. Ntsaluba. "National HIV and Syphilis Sero-Prevalence Survey of women attending Public Antenatal Clinics in South Africa 2000." Tech. rep., Directorate: Health Systems Research, Research Coordination and Epidemiology 2002, Department of Health, 2000.

[11] S. Armstrong, C. Fontaine, and A. Wilson. "2004 Report on the global AIDS epididemic.", 2004. URL `http://www.unaids.org/bangkok2004/report.html`.

[12] S. Allen. "Human immunodeficiency virus infection in urban Rwanda. Demographic and behavioral correlates in a representative sample of childbearing women." *The Journal of the American Medical Association*, vol. 266, no. 12, September 1991.

[13] W. Siriwasin. "HIV Prevalence, Risk, and Partner Serodiscordance Among Pregnant Women in Bangkok." *The Journal of the American Medical Association*, vol. 280, no. 1, pp. 49–54, July 1998.

[14] J. G. Ayisi, A. M. van Eijk, F. O. ter Kuile, M. S. Kolczak, J. A. Otieno, A. O. Misore, P. A. Kager, R. W. Stektee, and B. L. Nahlen. "Risk Factors for HIV infection among asymptomatic pregnant women attending an antenatal clinic in western Kenya Risk Factors for HIV infection among asymptomatic pregnant women attending an antenatal clinic in western Kenya." *International Journal of STD & AIDS*, vol. 11, no. 6, pp. 393–401, June 2000.

[15] M. Lallemant, S. C. Le-Lallemant, D. Cheynier, S. Nzingoula, G. Jourdain, M. Sinet, M. Dazza, and B. Larouze. "Characteristics associated with HIV-1 infection in pregnant women in Brazzaville, Congo." *Journal of Acquired Immune Deficiency Syndrome*, vol. 5, no. 3, pp. 279–285, 1992.

[16] S. Hassig. "Prevention of perinatal HIV transmission: are there alternatives to pre-pregnancy serological screening in Kinshasa, Zaire?" *AIDS*, vol. 9, pp. 913–916, September 1990.

[17] C. W. Lee and J.-A. Park. "Assessment of HIV/AIDS-related health performance using an artificial neural network." *Information & Management*, vol. 38, no. 4, pp. 231–238, February 2001.

[18] N. K. Kwak and C. Lee. "A Neural Network Application to Classification of Health Status of HIV/AIDS Patient." *Journal of Medical Systems*, vol. 21, no. 2, 1997.

[19] M. Abdella and T. Marwala. "The Use of Genetic Algorithms and Neural Networks to Approximate Missing Data in Databases." In *Proceedings of the*

IEEE International Conference on Computational Cybernetics, pp. 1001–1013. April 2005.

[20] M. S. K. Hornik and H. White. "Multilayer Feedforward Networks are Universal Approximators." *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[21] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford: Oxford University Press, 1995.

[22] J. Lampinen and A. Vehtari. "Bayesian Approach for Neural Nerworks - Review and Case Studies." *Neural Networks*, vol. 14, no. 3, pp. 7–24, April 2001.

[23] C. Bishop. "Bayesian Neural Networks." *Journal of the Brazilian Computer Society*, vol. 4, no. 1, July 1997.

[24] R. Neal. *Bayesian Learning for Neural Networks*. Ph.D. thesis, Graduate Department of Computer Science Toronto, March 1995.

[25] D. J. MacKay. *Bayesian Methods for Adaptive Models*. Ph.D. thesis, California Institute of Technology, Pasadena, California, U.S.A, December 1992.

[26] D. MacKay. "Probable networks and plausible predictions - a review of practical Bayesian methods for supervised neural networks." *Network: Computation in Neural Systems*, vol. 6, pp. 469–505, 1995.

[27] I. Nabney. *Netlab: Algorithms for Pattern Recognition*. Springer Verlag, 2003.

[28] W. Penny and S. Roberts. "Bayesian neural networks for classification: how useful is the evidence framework?" *Neural Networks*, vol. 12, pp. 877–892, 1999.

[29] D. J. MacKay. "The Evidence Framework applied to Classification Networks." *Neural Computation*, vol. 4, no. 5, pp. 720–736, 1992.

[30] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Berlin: Springer-Verlag, 1996.

[31] L. Davis. *Genetic Algorithms and Simulated Annealing*. London: Pitman Publishing, 1987.

[32] L. H. Tsoukalas and R. E. Uhrig. *Fuzzy and Neural Approaches in Engineering*. New York,: John Wiley and Sons, Inc., 1997.

[33] C. R. Houck, J. A. Joines, and M. G. Kay. "A Genetic Algorithm for Function Optimization: A Matlab Implementation." Tech. rep., NCSU-IE TR 95-09, 1995.

[34] J. Kennedy and R. Eberhart. "Particle Swarm Optimization." In *IEEE International Conference on Neural Networks*, vol. 4. IEEE Service Center, Piscataway Piscataway, NJ, 1995.

[35] R. C. Eberhart and Y. Shi. "Comparison between Genetic Algorithms and Particle Swarm Optimization." Tech. rep., Indiana University Purdue University Indianapolis, 2000.

[36] G. P. Zhang. "Neural Networks for Classification: A Survey." *IEEE Transactions on Systems, Man and Cybernetics-Part C: Applications and Reviews*, vol. 30, pp. 451–462, 2000.

[37] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Interscience, 2003.

[38] D. L. Hudson and M. E. Cohen. *Neural Networks and Artificial Intelligence for Biomedical Engineering*. IEEE Press Series in Biomedical Engineering. Piscataway, NJ: IEEE Press, 2000.

[39] D. B. Rubin. *Multiple Imputation for nonresponse in surveys*. New York: John Wiley and Sons, Inc., 1987.

[40] R. L. Rubin and D. Rubin. *Statistical Analysis with missing data*. New York: John Wiley and Sons, Inc., 1987.

[41] A. Ntsaluba. "Final Protocol, Annual Antenatal HIV and Syphilis Seroprevalence Survey." Tech. rep., Department of Health, 2000.

[42] P. Lisboa, A. Vellido, and H. Wong. "Bias reduction in skewed binary classification with Bayesian Neural Networks." *Neural Networks*, vol. 13, pp. 407–410, 2000.

[43] J. Snyman. *Practical Mathematical Optimization, An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms*. New Jersey: Prentice-Hall, Inc, 2005.

[44] A. P. Bradley. "The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms." *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[45] R. Neal. "Bayesian Training of Backpropagation Networks by Hybrid Monte Carlo Method." Tech. rep., Connectionist Research Group, Department of Computer Science, University og Toronto, 1992.

[46] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. "An Introduction to MCMC for Machine Learning." *Machine Learning*, vol. 50, pp. 5–43, 2003.

[47] C. Onwubiko. *Introduction to Engineering Design Optimization*. New Jersey: Prentice-Hall, Inc, 2000.

[48] G. James. *Advanced Modern Engineering Mathematics*. England: Prentice-Hall, Inc, 1999.

[49] R. Fletcher. *Practical Methods of Optimization*. England: John Wiley and Sons, Inc., 2003.

[50] S. Kirkpatrick, J. C. D. Gelatt, and M. Vecchi. "Optimization by Simulated Annealing." *Science*, vol. 220, no. 4598, pp. 671–681, May 1983.

[51] W. Bollweg and H. Maurer. "Numerical Prediction of Crystal Structures by Simulated Annealing." In I. M. Bomze, T. Csendes, R. Horst, and P. M. Pardalos, editors, *Developments in Global Optimization*, pp. 253–288. The Netherlands: Kluwer Academic Publishers, 1997.

Appendix A

Maximum Likelihood Neural Network Architecture Design

Training

The scaled conjugate gradient optimisation technique is used in error back-propagation to train the networks.

Number of Hidden Nodes

With the number of input nodes fixed at 12, the upper limit on the number of hidden nodes was 23, using a factor of 5 for the weights to data points ratio. However, the ability of the neural network to model the data depends on the nature of the data, the types of inputs and not just the quantity of training examples available.

Several networks of differing complexity between 0 and 23 hidden nodes were trained with a regularisation coefficient of 0.01. The networks were selected when a minimum validation error was reached, in the region where the validation error remained approximately constant. This method differs from early stopping, since it was ensured that the network was trained to a constant error, but the training was stopped at the lowest validation error. Due to random initialisation of the weights, the average of 5 runs was used to assess the trend.

Weight Decay Constant

The effect of α on validation error is that an increase in α results in the weights being restricted and hence regularisation of the training. Using the smallest network that consistently produced the lowest training and validation errors, alpha was optimised.

The best performing network was selected based on the area under the ROC curve for training and validation. The value of α is adjusted to ensure that the validation error decreases as much as possible, before reaching an approximate constant value. In contrast, with the early stopping method of regularisation, there is little or no weight decay, and the network is trained until the validation error starts to increase.

The optimisation of the regularisation coefficient requires that the validation error is observed as training progresses. With greater complexity(more hidden nodes), a larger α was required to achieve the same decrease in validation error.

Threshold adjustment

The output is binary: 1 for HIV positive and 0 for HIV negative. The result from the neural network, however, is a continuous value between 0 and 1 and this needed to be hard-limited to either 0 or 1. This was achieved by rounding the output to 1 if greater than a threshold and rounding it to 0 otherwise. The confusion matrix was calculated initially for a threshold of 0.5 on the training data, and this value was adjusted until the ratio of false positives to false negatives was approximately 1. Since the data set consisted of equal positive and negative outcomes, the classifier should produce equal false positive and false negative results, such that it is not biased toward predicting either case. The networks were optimised for a unity false positive to false negative ratio. The threshold was adjusted on the training data to ensure that there was no bias to either outcome, i.e. specificity and sensitivity have equal importance. Using this adjusted threshold, validation and final testing are limited at 0 or 1. It is incorrect to adjust the threshold on the validation set, since this set is not balanced.

One hundred different neural networks were trained and the best classifier on validation data was selected based on the area under the ROC . The best performing MLP network has a validation ROC of 0.6648. With an initial threshold of 0.5, the accuracy is 64.48% on training, but the number of false negatives (291) exceeds the number of false positives(212). The number of true positives is 417 and there are 496 true negatives. Shifting the threshold to 0.53 results in a better balanced classifier, with 250 false negatives and 258 false positives.

The best performing RBF network has a validation ROC area of 0.6319. The initial threshold gives 276 false negatives and 246 false positives, so the threshold is shifted to 0.51, resulting in 257 false negatives and 266 false positives on training. There are 22 hidden nodes and the activation function is thin plate spline.

Results

Maximum likelihood approach

The performance comparison is based on classification accuracy and training times. Although the RBF trains much quicker in 1.975744s than the MLP in 12.197473s, the accuracy on the RBF was significantly lower than was achieved for the best MLP network.

A summary of results for the simple networks is given in Table A1.

Table A1: Comparison of MLP and RBF simple classifiers

	MLP		RBF	
Conf. mat.	Pred. Pos.	Pred. Neg.	Pred. Pos.	Pred. Neg.
Actual Pos.	436	281	441	276
Actual Neg	106	204	126	184

Accuracy is calculated in two ways, the simple accuracy is the total correct results expressed as a percentage of total entries, the results are shown in Table A2.

$$SimpleAcc = \frac{TP + TN}{TP + FN + FP + TN} \tag{A1}$$

and

$$GeometricAcc = \sqrt{\frac{TP \times TN}{(TP + FN) \times (FP + TN)}} \tag{A2}$$

Table A2: Comparison of accuracy on MLP and RBF simple classifiers

Type	MLP	RBF
Simple	62.32	60.86
Geometric	63.26	60.42

The ROC curves are illustrated in Figure A1.

The results for the auto-encoders are in Figure A3.

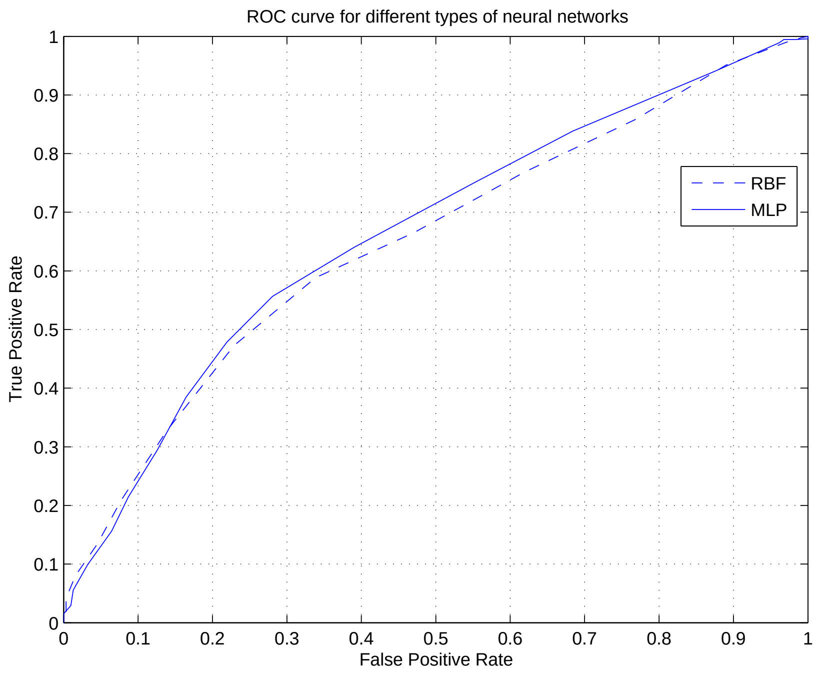

Figure A1: ROC curves for ANNs trained with maximum likelihood techniques

Table A3: Comparison of MLP and RBF auto-encoders

	MLP		RBF	
Conf. mat.	Pred. Pos.	Pred. Neg.	Pred. Pos.	Pred. Neg.
Actual Pos.	81	229	95	215
Actual Neg	146	571	186	531

The ability to classify true positive entries as a percentage of total positive entries is 26.13% on the MLP and 30.63% for the RBF. The classifier is clearly biased toward predicting negative entries, with MLP producing 79.64%, and 74.06% accuracy on the RBF. Improvement on theses results requires that the auto-encoder be trained to lower errors, and a larger set of negative entries for training.

Bayesian training of neural networks

Different permutations of number of hidden nodes and training cycles were simulated and the error assessed to determine the optimal parameters for each type of architecture as well as the training method. A summary of the average for 5 runs of each configuration are presented in Tables A4, A5, A6 and A7. Although the area under the ROC curve gives a good indication of classification ability, the areas are often similar. The curvature is also used to measure performance. The ROC curves

Table A4: Simulated results for MLP trained with evidence framework

No. of hidden nodes	No. of training cycles	AUC	Train Time (s)	Execution Time (s)	Error (%)
3	100	0.6528	21.422	0.406	0.410906
4	100	0.6523	25.938	0.516	0.400194
5	100	0.6332	30.751	0.688	0.413827
6	100	0.6238	35.313	0.875	0.416748
8	100	0.6124	47.203	1.297	0.433301
4	40	0.6479	13.687	0.531	0.413827
4	60	0.6435	17.812	0.531	0.41481
4	80	0.6495	21.793	0.531	0.403116

are shown in Figure A2. Since the training data has equal probability distributions of each class, the expected optimal threshold is 0.5, and the accuracy attained for this threshold indicates performance. Since we have used the naive Bayes method, the classification score or probability is uncalibrated, and the threshold must be determined from the data. For a calibrated classifier, the threshold is 1. Calibration is done by calculating the accuracy for each point, while tracing the curve, and returning the threshold with maximum accuracy. This process is unnecessary if the required sensitivity is known, and the aim is then to maximise the specificity, known as the Neyman Pearson method [44].

Evidence Framework

The best performing MLP has 4 hidden nodes, trained using 100 cycles with the scaled conjugate gradient optimisation technique. The hidden activation function is hyperbolic tangent, and the outer activation function is logistic, as was the case for the network trained with maximum likelihood. The error in Table A4 is the percentage error using a threshold of 0.5. The optimal threshold was found to be 0.53, determined by examining the ROC for training data. The trade-off is between sensitivity and specificity, allocating equal importance to both these properties is achieved by selecting the operating point nearest the point (0,1). The optimal RBF architecture has 25 nodes, trained for 100 cycles using an initial α and β of 0.01. The best performing function is thin plate spline. To achieve similar accuracy, the

Table A5: Simulated results for RBF trained with evidence framework

No. of hidden nodes	No. of training cycles	AUC	Train Time (s)	Execution Time (s)	Error 0.5 thresh (%)
15	100	0.6528	21.422	0.406	0.41091
20	100	0.6254	40.453	0.781	0.4275
25	100	0.6379	74.609	0.906	0.40701
30	100	0.6449	84.265	1.047	0.4148
25	20	0.5407	51.875	0.891	0.47030
25	40	0.6398	97.969	0.922	0.40798
25	60	0.6354	57.11	0.953	0.43428

radial basis function requires more nodes than the MLP, as was noted when training using the maximum likelihood approach. The training and execution times for RBF and MLP are quite similar, and as expected, is proportional to the complexity of the network. The accuracy is very similar for both architectures, although there is a significant difference in the area under the ROC curves, thus although a higher accuracy was obtained for the RBF, the MLP is once again a better classifier, and performs better over a wider range of threshold values. It is possible that a sub-optimal operating point was selected for the MLP, hence the slightly poorer accuracy. It is difficult to compare the different architectures though, since it is assumed that the comparison is between the optimal solutions for both types of networks.

The training times are shorter for similar sized RBF networks than for the MLP networks, once again, due to the two stage training procedure. However, to achieve similar size errors, the RBF networks have more hidden nodes, and thus the training procedure is longer for the RBF networks than it is for MLP networks.

Markov Chain Monte Carlo Training

The optimal MLP trained with the MCMC procedure has 5 hidden nodes, trained using 1600 samples with 100 omitted, 2000 steps in the trajectory and a persistence and variance of 0.001. The area under the curve is approximately equal to that obtained for the MLP trained with the evidence framework, implying that the classification performance is also similar. However, the final optimal accuracy is slightly

Table A6: Simulated results for MLP trained with MCMC sampling

No. of hidden nodes	AUC	Train Time (s)	Execution Time (s)	Error (%)
4	0.6477	4.75	2.031	0.4431
5	0.6513	9.922	4.015	0.4138
6	0.6381	8.687	4.172	0.4304
7	0.6318	9.422	5.141	0.4330
8	0.6125	10.453	4.875	0.4275

less than the MLP trained with the evidence approach. This is due to the selection of the threshold, illustrating that the area under the curve is a better measure of classification performance. The training time is much less than for the evidence approach, but the time taken to execute is much greater in this case.

Hybrid Monte Carlo Training

The optimal MLP trained with the HMC procedure has 4 hidden nodes, trained using 1000 samples with 100 omitted, 100 steps in the trajectory, a stepsize of 0.002.

Table A7: Simulated results for MLP trained with HMC sampling

No. of hidden nodes	AUC	Train Time (s)	Execution Time (s)	Error (%)
4	0.6520	304.265	2.188	0.4330
5	0.6469	342.437	2.703	0.4303
6	0.6369	630.312	2.719	0.4206
7	0.6468	645.032	2.703	0.4197
8	0.6228	721.563	3.281	0.4323

The training times are approximately 60 times longer than the MCMC method, while the execution times are similar. It is expected that the execution times would be similar, since the implementation requires a forward pass through the networks with the different weight samples. The advantage of the HMC method over the

MCMC method is the use of the gradient to ensure that the weights are generated according to the posterior probability distribution, and this should result in shorter training times, since less samples are rejected. The longer training times observed for the HMC implementation are due to the smaller step size. Many more steps were needed in the MCMC method to generate a representation of the probability distribution, due to the random walk implementation. The HMC algorithm should require shorter training times than the MCMC algorithm, but this was not evident in testing, and could be because only reasonably small networks were implemented.

The classification performance illustrated in the ROC curves in Figure A2 show that all neural networks trained with the Bayesian framework are very similar. The greatest area is produced for the MCMC trained MLP, and the ROC curve is also smoother with greater convexity. Noticeable local concavities are observed on the RBF network, indicating random performance in those regions. Although the area is greater than 0.5, indicating some classification ability, the ROC curves are less than ideal. An ideal ROC response should follow from the point (1; 1) along the line $y = 1$, but the actual response diverges from (1; 1) and most points are concentrated around (0,5; 0,5). This ties in with the results obtained when investigating the auto-encoders which were unable to detect differences in entries belonging to different classes. In comparison to the ROC curves for the standard neural networks, the Bayesian trained networks are much more convex. The standard ROC curves have pronounced knee in the curvature, indicating the the optimal operating region is much smaller than that of the Bayesian trained networks.

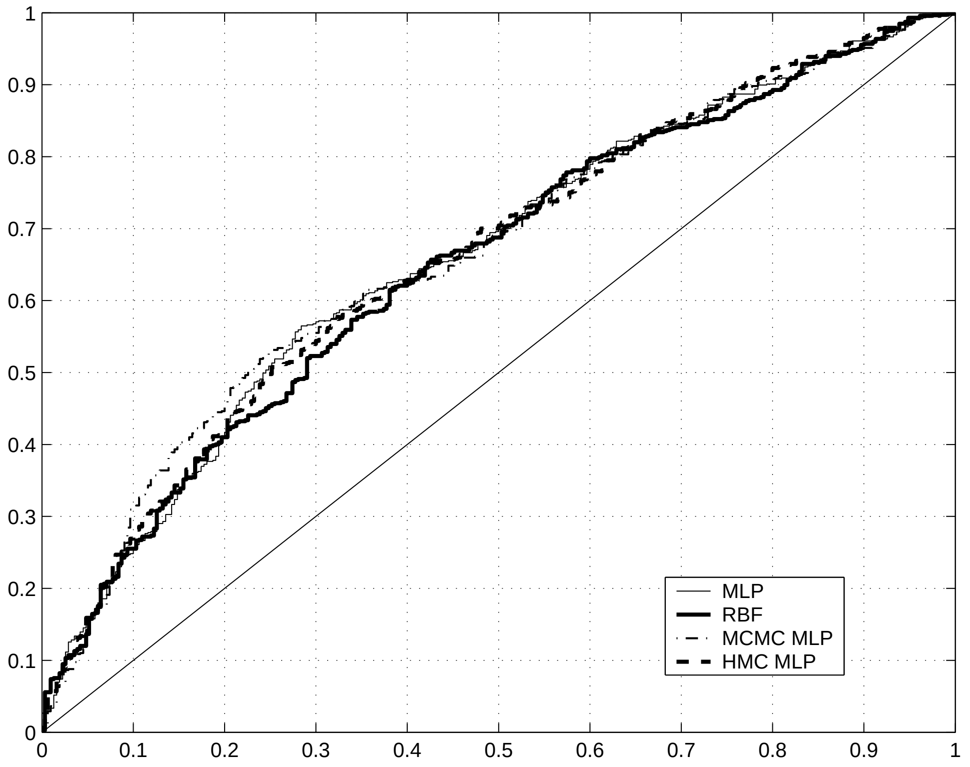

Figure A2: ROC curve for Bayesian neural networks

Appendix B

Markov Chain Monte Carlo Methods

Bayesian methods are computationally intensive due to the integration over multi-dimensional spaces. Integration is required for the analytical evaluation of the evidences for example, and to use the property of marginalisation, which is a method to eliminate variables through integration. The Gaussian approximations allow the integrals to be performed analytically, but in other cases the integration is usually mathematically intractable. In such cases, the integral is replaced with a finite sum of randomly sampled points over weight space.

The integrals to be evaluated take the general form:

$$I = \int F(w)p(w|D)dw \tag{B1}$$

where p(w—D) represents the posterior distribution of the weights, and F(w) is some integrand. This integral is approximated by a finite sum in the form

$$I \approx \frac{1}{L}\sum_{i=1} LF(w_i) \tag{B2}$$

where w_i is a sample of weight vectors generated from the distribution $p(w|D)$.

The Markov Chain Monte Carlo methods generates the random weight vectors from a sequence, where each successive vector depends on the previous vector plus a random component. The simplest method is a *random walk*.

$$w_{new} = w_{old} + \epsilon \tag{B3}$$

where ϵ is a small random vector.

Since the vectors depend on the previous vectors, they are no longer independent. The goal is to produce vectors representative of the posterior distribution of weights, and these chains of vectors can be produced by Metropolis, Gibbs, and Independence chains. In Markov chain generation, the design parameters are the number of iterations, and the number of observed values after convergence. In this study, vectors are chosen probabilistically according to the Metropolis algorithm. Weight vectors that result in an increase in the posterior distribution will be accepted according to the algorithm, and those resulting in a decrease can be accepted, but with probability lower than 1. The criteria for acceptance are as follows:

```
if $p(w_{new}|D) > p(w_{old}|D)$
then accept
else
if $p(w_{new}|D) < p(w_{old}|D)$
then accept with
probability $\frac{p(w_{new}|D)}{p(w_{old}|D)}$
end
```

Although the Metropolis Algorithm is an improvement on the simple Monte Carlo method, it is still slow for complex networks [45]. In order to achieve reasonable acceptance probability, the changes must be small, so a large number of iterations are required to reach and explore the distribution.

Hybrid Monte Carlo Methods (HMC)

HMC is a modified Monte Carlo method, which makes efficient use of gradient information to reduce random walk behaviour, speeding up exploration of the weight search space [46]. The gradient indicates in which direction one should go to find states with high probability. The HMC uses a Hamiltonian function, which describes the conservation of energy:

$$H(q,p) = E(q) + \frac{1}{2}|p|^2 \tag{B4}$$

The dynamic moves producing the Markov Chain are obtained by differentiating q and p with respect to "time" as follows.

$$\frac{dq}{dt} = \frac{\partial H}{\partial p} = p \tag{B5}$$

$$\frac{dp}{dt} = -\frac{\partial H}{\partial q} = -\nabla E(q) \tag{B6}$$

Properties of Hamiltonian dynamics are time reversibility and Liouville's theorem, which states that a volume of region space does not change as it evolves, and this ensures the validity of the procedure [45].

The equations can be discretised with a non zero step size *epsilon*, using the "leapfrog" method.

$$p(t + \frac{\epsilon}{2}) = p(t) - \frac{\epsilon}{2}\nabla E(q(t)) \tag{B7}$$

$$q(t + \epsilon) = q(t) + \epsilon p(t + \frac{\epsilon}{2}) \tag{B8}$$

$$p(t + \epsilon) = p(t + \frac{epsilon}{2}) - \frac{\epsilon}{2}\nabla E(q(t + \epsilon)) \tag{B9}$$

The algorithm constitutes the iteration of equations (B7) to (B9), and the number of iterations is L. These L steps are used to generate candidate weights. The selection of both L and ϵ involves trade offs: large values of ϵ result in low acceptance rates, while small values require many leapfrog steps. The the aim in the optimisation of L is to generate candidates far from the initial state, but near enough for the computation to be viable [46].

The Hybrid Monte Carlo method can implement other methods of optimisation of the energy state H, such as simulated annealing, an optimisation method based on statistical mechanics [45], or Gibbs sampling.

Appendix C

Gradient Based Methods

All gradient based methods require an analytic calculation of the gradient. A search direction and step-size provide information about the next solution, this is seen as travelling in the search space along the surface of solutions. The major downfall in gradient based methods is that since the calculation of the gradient is local, it is prone to falling into local minima. This is particularly evident in the gradient descent method.

Multivariable optimisation can be seen as error surface minimisation. These algorithms are line search descent. The basis for gradient methods is that the fastest way to locate a minimum is to move along the gradient, which is a vector of directional derivatives of the error function.[47]

The general algorithm is an iterative process: starting at an initial estimate to the optimum point x(0), and moving from point x(n)to point x(n+1) along a line, as shown in equation C1.

$$x_{n+1} = x_n + \lambda_n \tilde{d}_n \tag{C1}$$

Where d is the search direction, and λ is the step size.

The general algorithm is as follows[43]:

1. Given a starting point x_0 and tolerances (stopping criteria), set n = 1
2. Compute search direction d_n
3. Perform a one-dimensional line search in direction d_n to give the step length λ_n
4. Update the design variables

5. Test for convergence, based on either change in x, gradient, or value of objective function

6. n = n+1

The gradient methods differ from each other in the way that the search direction and step length are calculated. The first order methods, such as gradient descent and conjugate gradient methods use first derivative information only. Quasi-Newton and the secant method also require second derivative information.

Gradient Descent

This method is the simplest of the gradient methods since the search direction at each iteration is simply the negative gradient at that point. The negative gradient is selected as it will decrease the function value by the largest amount. The equation for the method is:

$$x_{n+1} = x_n + \lambda_n \nabla f(x_n) + \mu(x_n - x_{n_1}) \tag{C2}$$

A critical factor of the gradient descent method is the step size. If the step size is too large, from one iteration to the next, the solution may move backward and forward over the minimum. If the step size is too small, the solution may never converge. The step size can be selected by a line search, demonstrated in equation C3.

$$f(\lambda_n) = f(x_{n-1} + \lambda_n d_n) = min_\lambda f(x_{n-1} + \lambda d_n) \tag{C3}$$

Another weakness is that since gradient is evaluated locally, if the curvature of f (given by the Hessian) varies significantly with direction (i.e. near a minimum), then the gradient will not point to the minimum.[27]

To allow the solution to escape local minima, the optional momentum term is added to the formula.

The characteristic zig-zagging, or oscillatory behaviour is due to the fact that consecutive steepest descent directions are *orthogonal*[43]. As a result, the gradient descent method is typically slow to converge, particularly when the problem is poorly scaled. The rate of convergence is linear, but can be quadratic for well scaled problems. Although there is a theoretical proof of convergence, the method usually terminates far from the solution due to round-off effects.[43] It can also converge to a saddle point.

The advantage to this method is its computational simplicity, but the slow convergence, particularly near the optimum means that it is rarely used[47; 48].

Conjugate Gradient

The search directions are the result of orthogonalization of the successive gradients.

The initial point is at x_0, and direction d_1 given by

$$d_1 = -g(x_0) \tag{C4}$$

The initial search direction is the negative gradient as for the gradient descent method: $g_n = -\nabla f(x_n)$.

$$x_{n+1} = x_n + \lambda_n d_n \tag{C5}$$

The new direction vector is the basis of the Polak and Ribiere algorithm, developed from the method proposed by Hestenes and Stiefel [49; 47]

$$d_{n+1} = -g_n + \beta_n d_n \tag{C6}$$

where β_n

$$\beta_n = \frac{(g_{n+1} - g_n)^T (g_{n+1})}{g_n^T g_n} \tag{C7}$$

Substituting for g and d:

The new direction vector is

$$\nabla f(x_{n+1}) = \nabla f(x_n) - \beta_n \nabla f(x_n) \tag{C8}$$

where β_n

$$\beta_n = \frac{(\nabla f(x_{n+1} - \nabla f(x_n)))^T \nabla f(x_{n+1})}{\nabla f(x_n)^T \nabla f(x_n)} \tag{C9}$$

The advantages of the conjugate gradient method are that it eliminates the need to calculate and store the Hessian matrix (as must be done for quasi-Newton methods). Secondly, since the conjugate directions are not orthogonal to each other, their use is an improvement on the steepest descent method where the two consecutive search directions are orthogonal, causing slow convergence. The disadvantage is that calculation of the line search requires several additional function evaluations per step[27].

Scaled Conjugate Gradient

The Scaled Conjugate Gradient technique improves on the Conjugate Gradient as there is no line search required. The step size is calculated using a formula developed by Moller[27]. This uses the Hessian, which is approximated using a finite difference method for computational efficiency.

Quasi Newton

Compared to the first order methods, the Newton methods have a rapid convergence, due to the inclusion of second order information. However, the main disadvantage is more computation at each iteration.

Newtons method uses a second order Taylor series expansion of $f(x)$ about the nth point. It finds the minimum of this quadratic approximation, and and uses this value to start the cycle again.[48]

$$x_{n+1} = x_n - H_{-1}(x_n \nabla f(x_n) \tag{C10}$$

where $\nabla f(x_n) = \frac{\partial f(\widetilde{x_n})}{\partial \widetilde{x}}$, $H = \frac{\partial^2 f(\widetilde{x})}{\partial x_i \partial x_j}$

Although the Newton method is quadratically convergent near the solution, there are many other disadvantages. One of the main difficulties is that the Hessian and its inverse must be calculated and stored at each iteration, which is very computationally inefficient. In addition, formulae from which the second derivative can be evaluated must be supplied. The method is not always convergent, even if x_0 is close to the solution.

The quasi-Newton avoids computing the actual Hessian by updating an initial positive matrix (usually the identity matrix) at each iteration. Only the first derivatives required, and $O(n^2)$ multiplications per iteration $O(n^3)$ The storage requirement is

$O(n^2)$, which is the same as for Newton's method, and more than the space required by the conjugate methods [27]. The advantage over conjugate gradient methods is there is no need to periodically restart iterations[49].

Convergence is better than for the Newton method as the Hessian, is forced to be symmetric and positive definite. The most widely used and efficient formulating the BFGS method

Typically, quasi-Newton methods are used for problems with less than a hundred variables[27].

Secant

The secant method is based on the Newton method and the quasi-Newton method, but approximates both the Hessian and the gradient. This gradient approximation is useful if the gradient is difficult to calculate, or impossible to obtain. One method of approximating the gradient is the central difference method.

Appendix D

Gradient Based Methods

Two issues are investigated:

1. The performance of the different gradient methods for solving a single variable problem (on single variable missing) and for solving a multivariable problem (multiple variables missing).
2. The effect of the number of missing variables on the accuracy of prediction

The performance of different methods

To evaluate the optimisation methods, it is necessary to estimate the same data used to train the neural network(as opposed to using the test set). In this way, the performance of the algorithms can be measured, regardless of the predictive ability of the network. This is equivalent to comparing the error obtained by substituting the final solution into the error function. The variables are scaled, so the error would give no indication of the error of the scaled values. Instead of analysing the scaled error, the scaled solutions for the missing values obtained by the different methods are compared. The same network is used to compare results thus this analysis is a valid way of measuring the efficiency of the algorithms.

The actual performance of the system must be measured using unseen test data, to check the predictive ability of the network and the data estimation ability of the system. These two approaches were taken in assessing the performance on both single variable missing and multivariable missing cases. Testing on the multiple missing variable case is necessary to determine the most efficient method to be used in part 2 of the testing(effect of number of missing variables).

Table D1: Performance on 1 missing variable for training data

initial	Grad. Desc.	Conj. Grad	SCG	Quasi New.	Secant	Value
13	18.6368	18.6368	18.5120	18.5120	18.6368	18
65	18.7460	18.7460	18.9384	18.7460	18.7460	18
61	18.6368	18.6368	18.8292	18.6784	18.6368	18
37	18.6420	18.6420	18.7252	18.6420	18.6420	18
35	18.7252	18.7252	18.9228	18.9228	18.7252	18

Table D2: Performance on 4 missing variables for training data

initial	Grad. Desc.	Conj. Grad	SCG	Quasi New.	Secant	Value
23.545	14.066	14.196	14.097	13.956	13.780	16
2.583	3.526	3.641	3.621	3.419	3.499	7
7.245	0.837	0.849	0.831	0.826	0.788	1
27.154	16.104	16.255	16.120	15.964	15.761	18

One aspect of this test is what effect the *initial value* has on the solution, as shown in Table D1, the system appears to be insensitive to the initial value. This is because the objective function is smooth, so there are few or no local minima. Five different initial values were used: minimum (0), maximum (1), and 3 random values between 0 and 1.

On a multivariable problem, only one case of initial values is shown in Table D2.

The time taken to converge is an effective performance measure, but must be viewed in conjunction with the error at convergence. Often, methods have short convergence times and appear more efficient, but actually converge at a much higher error than others.

Only the full time taken to converge is taken into account, as some methods require more iterations with shorter iteration times, thus neither iterations nor iteration time are effective performance metrics. From Table D3 it is clear that the secant method performs the best in terms of converging at the lowest error, as well as in the shortest time. The next best performing algorithm is the quasi-Newton method. This is to be expected, since both these methods are second order and utilise information about the curvature in the Hessian.

Although the Scaled Conjugate Gradient Method is often faster to converge, it usually converges at a higher error than the other methods. The worst performance

Table D3: Performance of different methods on 4 missing values for training data

Metric	Grad. Desc.	Conj. Grad	SCG	Quasi New.	Secant
Time	6.812	6.843	4.828	5.359	0.579
Error	1.178e-3	1.047e-3	1.208e-3	1.174e-3	1.040e-3
Time	1.844	1.266	0.61	1.437	0.688
Error	1.138e-4	1.135e-4	1.139e-4	1.155e-4	1.103e-4
Time	2.219	1.718	0.766	1.797	0.484
Error	9.300e-5	9.439e-5	9.252e-5	9.230e-5	9.050e-5
Time	8.5620	8.7490	3.4530	7.4530	1.063
Error	1.479e-4	1.465e-4	1.462e-4	1.460e-4	2.462e-5
Time	2.047	1.531	3.453	1.9060	0.594
Error	7.517e-3	1.385e-3	1.434e-3	1.453e-3	1.090e-3

is that of the gradient descent method, as the time taken to converge is the highest for this method. This relationship between error convergence with time is best illustrated in a graph, as seen in the figure below.

The computational efficiency of the algorithms is measured by counting the number of function and gradient evaluations, illustrated in Table D4. The Secant and Scaled Conjugate Gradient methods make the fewest function evaluations since they both approximate the Hessian. The gradient descent evaluates the gradient for each iteration requiring many iterations before convergence.

Table D4: Number of Function and Gradient Evaluations for one problem (entry)

Method	Grad. Desc.	Conj. Grad	SCG	Quasi New.	Secant
# Function	414	313	21	231	53
# Gradient	12	8	41	7	21

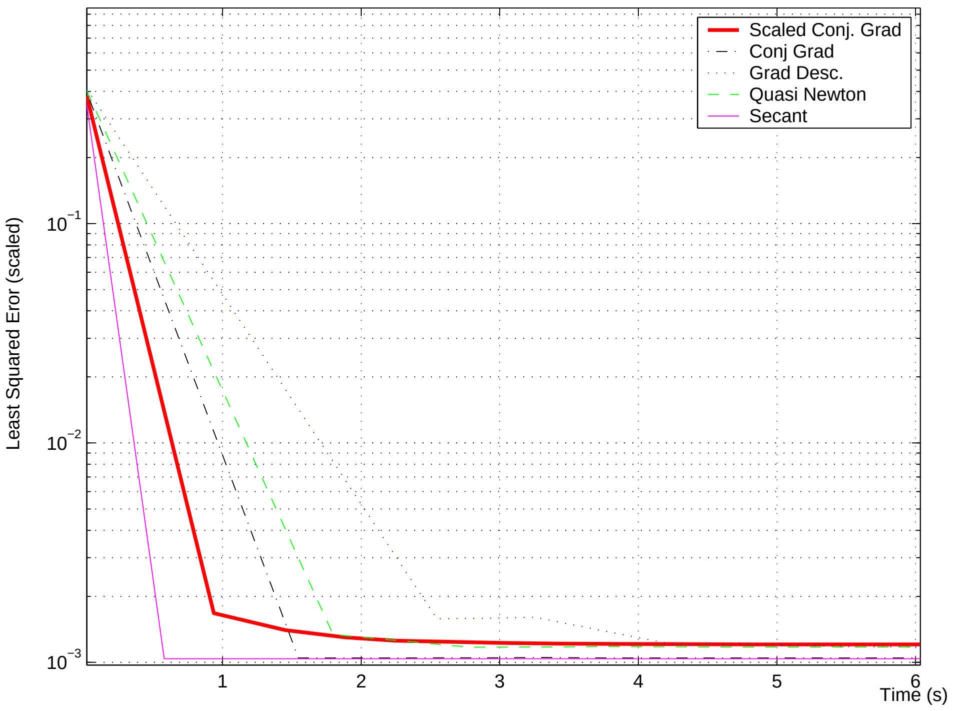

Figure D1: Comparison of convergence rate for different optimisation methods

Appendix E

Simulated Annealing

Background

The origins of simulated annealing lie in statistical physics, and was first applied to problems of combinatorial optimisation by Kirkpatrick *et al* in 1983[50]. It has since been applied to problems of computer design, image processing, combinatorial optimisation and artificial intelligence [31]. Statistical mechanics is the study of the behaviour of large systems of interacting components. In annealing, the system is composed of atoms in thermal equilibrium at a finite temperature. When referring to the behaviour of the system, this is taken to be the positions of the particles. The spatial positions of the components can be grouped as the configuration of the system, and this is dependent on the energy of the system. For a system in thermal equilibrium at a given temperature T, the probability $\pi_T(s)$ that the system is in a given configuration s depends on the energy $E(s)$ of the configuration, which obeys the Boltzmann distribution, defined in (E1).

$$\pi_T(s) = \frac{e^{\frac{-E(s)}{kT}}}{\sum_{\omega \in S} e^{\frac{-E(\omega)}{kT}}} \tag{E1}$$

where k is Boltzmann's constant and S is the set of all possible configurations.

Using a technique developed by Metropolis, a system of particles in thermal equilibrium at temperature T can be simulated as the ratio of the probability of the system being in the next configuration and the current configuration. A future configuration r (at time $t+1$)is selected according to the difference between the energy of r and the energy of the present configuration q (at time t). Using equation (E1), we obtain (E2).

$$p = \frac{\pi_T(r)}{\pi_T(q)} = e^{\frac{-(E(r)-E(q))}{kT}} \tag{E2}$$

Table E1: Simulated Annealing Optimisation analogues

Concept	Physics	Optimisation
Function	Energy	Objective
Configuration	Particles	Parameters
Aim	Low-energy config.	Optimal solution
Control parameter	Temperature	Temperature

A configuration r at $t+1$ is automatically accepted if its energy is strictly less than that of q at t, i.e. if $p > 1$. Configurations of higher energy can be obtained: if $p \leq 1$, the energy of r is greater than or equal to that of q, the configuration r is accepted with probability p. As $t \to \infty$, the probability that a system is in configuration s equals $\pi_T(s)$.

Definition

Simulated annealing is a computational technique for finding globally minimum cost solutions to large optimisation problems. Annealing is a thermal process for obtaining low energy states of a heated solid. In the process, the temperature of the system is first elevated and the gradually lowered, spending enough time at each temperature to reach thermal equilibrium. Spending insufficient time at each step reduces the probability of attaining a very low energy configuration. The optimisation analogues of the physics concepts are presented in Table E1. The simulated annealing process therefore contains three steps:

- Identify the analogues for the process
- Select an annealing schedule, which consists of a set of decreasing temperatures and the time spent at each temperature
- Design a method for generating and selecting different configurations

Algorithm

The pseudo code for the basic algorithm is from [51]. T is the initial temperature, q is the initial state, and M is the length of the Markov chains.

Parameters

The success of the algorithm depends on the suitable choice of the starting temperature T_0, choice of the neighbourhood S_q of the current state q, length of the Markov chains, specification of the cooling schedule $T_k \Rightarrow T_{k+1}$ and a stopping criterion.
Determination of a Starting Temperature: The starting temperature must be high enough to ensure that all possible states of the system can be reached, but low enough to minimise computation time. [51] suggests that in most cases, an appropriate starting temperature is related to an average value of the objective function.
Step-size and Neighbourhood: Variable step-size methods can be used if the objective function is not computationally expensive. If too many function evaluations are required to calculate a step-size for the present state, it is more efficient to use a fixed step-size.
Length of the Markov Chains: The length of the Markov Chains determines how long the algorithm remains at a fixed temperature [51]. Long Markov chains increase computational effort, while short ones prevent the algorithm from producing results with acceptable probability.

Advantages and Disadvantages

Simulated annealing eliminates most disadvantages of gradient based methods as the solution does not depend on initial value. The probability of acceptance of the new point depends on the temperature, which is another control parameter. On the otherhand, determining the proper annealing schedule (schedule of temperatures) is a matter of trial and error. The process is inherently slow, but this can be overcome by implementing the Simulated Annealing algorithm using parallel processing.

Appendix F

Evolutionary Methods

The performance of the different methods depends on the degree to which the parameters for each algorithm are optimised. For a fair comparison, one parameter, such as the number of function evaluations should remain constant. Time is also not necessarily a good indicator of performance, since the amount of processing by each algorithm depends on the parameters. It must be considered though, since one of the biggest trade-offs in the design of these optimisers is accuracy versus time. For example, a longer cooling schedule, more generations, or more iterations in the particle swarm all ensure more function evaluations and hence take longer, but this improves accuracy. Function evaluations and times are provided in Table F.2. Accuracy is the basis for comparison in this analysis. Each evolutionary algorithm is run 50 times on 1 data entries in the training set and consistency of accurate performance is illustrated by the histograms of error in Figures F.1, F.2 and F.3.

As shown in Table F.1, all methods reach the same optimum. For the purposes

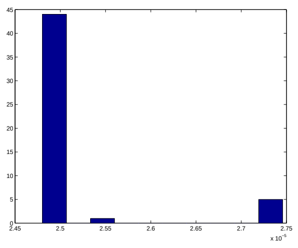

Figure F.1: Histogram of error for PSO

of approximating the data, since the values are rounded (the data is discrete) all methods produce acceptable results, so the time becomes the deciding factor.

5 different entries with 4 missing variables were optimised, and the convergence time with error, total execution time, and function evaluations are summarised in Table F.2. The error convergence with time characteristic is given in Figure F.4.

Table F.1: Performance on 1 entry of training data with 4 missing variables

Variable	Actual	GA	PSO	Sim. Ann	Hybrid
Mother age	29	24.6395	24.6373	24.6320	24.6542
Education	9	9.4746	9.4798	9.4902	9.4565
Gravidity	3	3.1158	3.1176	3.1272	3.1120
Father Age	30	38.1554	38.1593	38.1253	38.1688

Table F.2: Performance on 4 missing variables per entry (training data)

Metric	GA	PSO	Sim. Ann	Hybrid
convergence time	36.8462	10.03	65.688	12.719
error	2.4804e-5	2.4803e-5	2.4806e-5	2.4803e-5
function evals	13822	4280	30133	20916
execution time	40.0160	13.359	275.77	177.156
convergence time	37.2304	9.67	64.75	15.828
error	2.4195e-7	5.5332e-7	2.3824e-7	2.3525e-7
function evals	13979	4500	28907	21093
execution time	40.3280	13.281	264.52	175.781
convergence time	31.672	9.25	51.141	15.453
error	2.8656e-6	1.9012e-6	1.9018e-6	1.9012e-6
function evals	11336	4305	30231	20600
execution time	33.1720	13.281	276.53	173.84
convergence time	10.766	8.95	13.703	18.672
error	5.6786e-5	5.6786e-5	5.679e-5	5.6786e-6
function evals	11959	4170	28433	20993
execution time	34.8590	13.266	259.56	175.89
convergence time	34.078	11.343	31.797	43.844
error	7.805e-7	7.8032e-7	7.8244e-7	7.8172e-7
function evals	14259	5160	28222	20877
execution time	41.0470	13.25	261.28	176.36

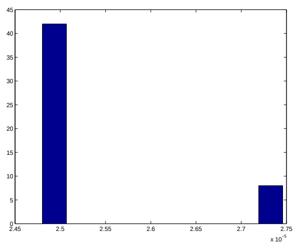

Figure F.2: Histogram of error for GA

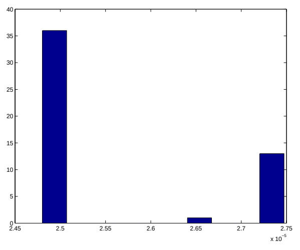

Figure F.3: Histogram of error for Simulated Annealing

Simulated Annealing

Consistent results were obtained for this method, but it is computationally intensive, and the execution time is greater than all the other methods. The major disadvantage of simulated annealing is that the optimisation is performed on one solution. In contrast, the GA and PSO optimise many solutions, and this speeds up the optimisation. To counteract the possibility of an incorrect initial temperature, several annealing runs were carried out. However, these are performed serially, hence the execution time is much longer than that of the other methods. In addition, each annealing run has a longer convergence time, due to the slow cooling schedule.

Genetic Algorithm

The mutation rate had a more pronounced effect on the accuracy than the crossover rate or population size. The precision of the solution is not as important as evaluating much of the search space, as due to the rounding of the solutions, precision is lost anyway.

Particle Swarm

The PSO success can be attributed to the group interaction and search space is effectively broken down into areas covered by each agent. Since the algorithms

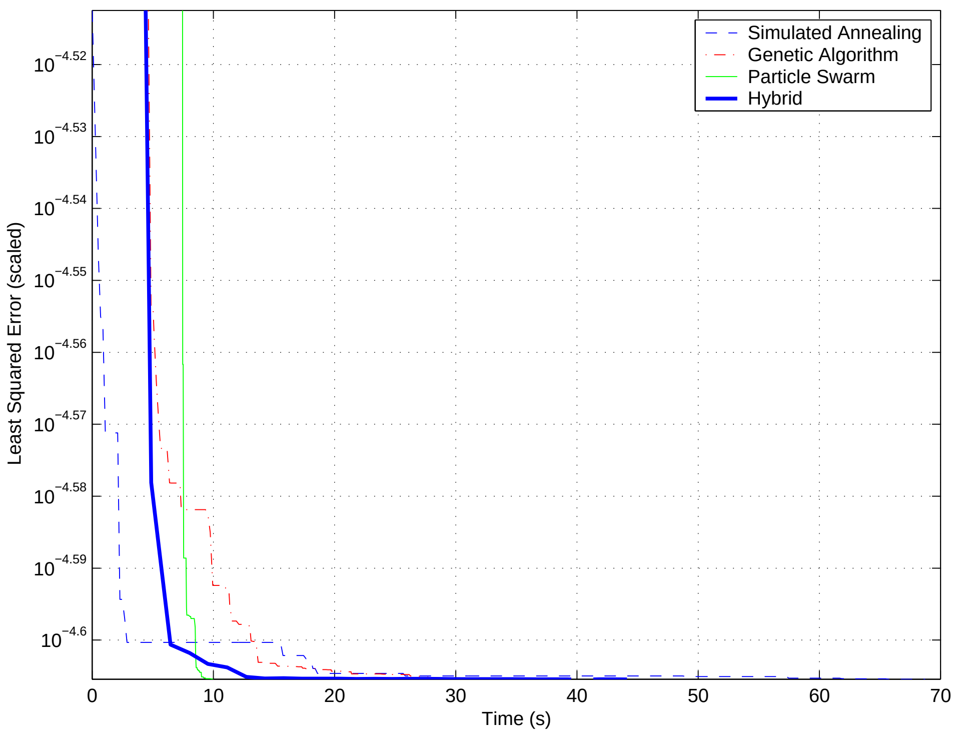

Figure F.4: Convergence rates for evolutionary and hybrid optimisation methods

produce similar results in terms of accuracy, the most efficient algorithm is the particle swarm optimiser, with an execution time of 35% of the time required by the GA. The second part of the testing is performed on unseen testing data, with varying number of variables missing.

Hybrid Simulated Annealing-Genetic Algorithm

The convergence rate per run is comparable to the PSO and as seen in Figure F.4, the convergence rate of the hybrid method is better than both the GA and the SA, proving that the hybrid method combines the merits of each individual method. Compared to the PSO, the hybrid method is also more robust, evident in the accuracy in the results. This robustness is due to the multiple runs executed in the main SA algorithm, whereas in the PSO only one run was executed. The multiple executions are also the cause of the lengthy execution times, and this can be reduced by having fewer runs. This hybrid method is not as dependant on the initial state as the original SA, so fewer runs can be justified. If accuracy is more important than the execution time, then the hybrid method is preferable to the PSO. If however there are time constraints, (as would be the case for large datasets), then the PSO is preferable, though the hybrid method can be implemented with a single run to save time.

Since the PSO is the fastest and most accurate method of the stochastic methods tested, better results could be achieved by combining PSO and the SA. The implementation would be similar to the hybrid implemented in this study: the PSO would be used to generate the samples, and the SA can be used to probabilistically select the following state.