An Investigation of Applications of Artificial Neural Networks in Medical Prognostics

Peter K. Ghavami

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington

2012

Reading Committee:

Kailash C. Kapur, Chair

Archis Ghate

Christina Mastrangelo

Norman Beauchamp

John Bramhall

Program Authorized to Offer Degree:

Industrial and Systems Engineering

University of Washington

Abstract

An Investigation of Applications of Artificial Neural Networks in Medical Prognostics

Peter K. Ghavami

Chair of the Supervisory Committee:
Professor Kailash C. Kapur
Industrial & Systems Engineering

During the course of care, patients frequently develop escalating health problems that lead to medical complications, costly treatments, severe pains, disabilities and even death. Predicting such escalations provides the opportunity to apply preventive measures that result in better patient safety, quality of care and lower medical costs; in short, timely prediction can save lives and avoid further medical complications. Prognostics methods using Artificial Neural Networks (ANN) promise to deliver new insights into future patient health status that provide more effective medical treatment during the patient hospital stay.

With the advent of smaller, inexpensive sensors and volume of data collected from patients, physicians are challenged with making increasingly analytical decisions from a large set of data that are being collected per patient. This trend is only increasing giving rise to what's known in the industry as the "big data problem": The rate of data accumulation is rising faster than physicians' cognitive capacity to analyze increasingly large data sets to make decisions. The big data problem offers an opportunity for predictive analytics and prognostics.

Investigation and development of a methodical framework for medical data prognostics in general and use of committee of algorithms in particular have not been adequately explored.

A framework for prediction of patient health status from clinical data is needed to assist physicians in their clinical decision process. This research investigates and contributes to three essential ideas for improving healthcare prognostics through big data analytics: 1) A control system approach to prognostics for prediction, 2) A generalized committee of models framework as prognostics engine, and 3) Study the viability of such framework on a particular clinical case.

This research offers three key contributions:

First, it develops a control system treatment of medical prognostics and predictive models. The control system development of prognostics combines feed-forward and feedback control mechanisms to create a framework for medical prognostics. This framework introduces a rules-based prognostics engine that uses ANN algorithms to identify patients who develop a particular disease or medical complication.

Second, it provides a generalized committee of models framework to predict the patient's medical condition and predict any medical complication from large data sets. The model also provides the strength (or the impact level) of all contributing clinical data to that prediction. The methodology proposes using a multi-algorithm prognostics framework to enhance the accuracy of prediction using four ANN models. The framework introduces a supervisory program, called an oracle to select the most appropriate ensemble of models that best meet the practitioner's desired prediction accuracy.

Third, it demonstrates the viability and feasibility of using ANN methods as predictive models in this framework. As part of the demonstration, the research explores building, training and validating four ANN models to predict medical complications from data acquired during 1,073 patients' hospital stay to predict Deep Vein Thrombosis/Pulmonary Embolism (DVT/PE). DVT/PE, is a condition caused by blockage of patient lung vessels by blood clots that initially form in patient's legs. DVT/PE leads to severe pain, loss of lung function and even death.

The aim of all three ideas is to improve the physician's ability to make predictive decisions from a vast array of data in order to be proactive and apply preventative medical interventions before complications occur.

Contents

LIST OF FIGURES

LIST OF TABLES

DEDICATION

To my dear wife Massi.

ACKNOWLEDGEMENTS

The author wishes to express sincere appreciation to the Department of Industrial and Systems Engineering and UW Medicine Faculty for their extended long term support and especially to Professors Kailash Kapur, Norman Beauchamp, John Bramhall and Michael Souter for their vast reserve of patience and knowledge. This thesis would never have been completed without the encouragement and devotion of these professors and my dear wife, Massi.

Chapter 1

1 INTRODUCTION

1.1 Motivation

Predicting medical conditions for patients during their hospital stay is regarded as one of the most challenging and rewarding undertakings for physicians when such predictions are timely and informative enough for medical intervention. During their course of care, patients frequently experience escalating health problems that lead to further medical complications. These complications, mostly regarded as preventable (Maguire 2007), cause severe pains, injuries, disabilities and even death among patients. Several studies have suggested that complications are common with estimates of frequency ranging from 40% to 95% (Davenport, Dennis, Wellwood, Warlow 2006), and some relate poor outcomes to such complications (Johnston, Lyden, Hanson, Feasby, et al 1999). Prediction and diagnosis of escalating medical conditions has been a province of Clinical Prediction Rules (CPRs). CPRs are a field of medical research in which researchers attempt to identify rules for prognostics or diagnoses of disease by scoring a combination of medical signs, symptoms and other physiological and clinical findings (Jervis, McGinn 2008).

CPR scores are determined by researchers using algorithms that are predominantly based on linear regression or similar linear statistical methods. Despite advances with CPRs, it has been shown that physicians encounter lower accuracy and generalizability when they apply CPRs in their practice (Toll, Janssen, Vergouwe, Moons 2008). The key reasons cited for poor accuracy include differences from the initial patient population that the rule was developed and the current patient under treatment. This reflects on the fact that CPR rules are not extensible and adaptive enough to adjust to these differences. An adaptive method is needed to make more specific prediction tailored to a specific population of interest.

In another research front, prediction has been a topic of study by engineers. To apply an engineering approach to prediction, interest in Prognostics and Health management (PHM) field has been growing (Pecht 2008). PHM is a discipline focused on predicting the time at which a component in a system will no longer perform as intended. When applied to medical prediction, PHM predicts when a physiological organ will fail or when a disease will occur. PHM is a relatively new field that promises to help medical prognostics (Ghavami, Kapur 2011). Among adaptive mathematical methods used for prognostics, Artificial Neural

Networks (ANNs) have become popular in the last two decades as powerful prediction tools. ANNs owe their popularity to their ability to model non-linear relationships, handle adaptive learning, pattern recognition and classification –features that can be helpful in building medical predictive models.

1.2 Problem Statement

Despite advances with Clinical Prediction Rules (CPRs), it has been shown that physicians encounter lower accuracy and generalizability when they apply CPRs in their practice (Toll, Janssen, Vergouwe, Moons 2008). The reasons for poor accuracy are generally attributed to differences between the patient data under treatment and the initial population that the rule was developed. These differences are due to geography (the region or site) differences, domain differences (age, sex or clinical specialty) and temporal differences (rules become less accurate "over time"). Physicians often find prediction quite challenging due to this difficulty plus four additional challenges: 1) not all input data required by the CPR may be available; 2) not knowing when a particular CPR rule is applicable, 3) A CPR does not exist, or the CPR score is not definitive, and 4) The research is contradictory or inconclusive.

The rise in sensor technology now affords us with more accurate and frequent data collection methods. Frequency of lab test results, diagnostic tests and even the genomics data that's becoming easier to obtain combined with advances in computer storage systems afford large sums of data accumulation per patient. This gives rise to a "big data problem" in medicine that provides both a challenge and opportunity. The challenge is that the data volumes are vast and are getting larger, so big that they exceed human cognition's limits for analysis. The opportunity is that new and diverse analytical tools can be applied to assist physicians in diagnosis and even prediction.

Yet the tools necessary to analyze such big data are still not fully standardized and remain exclusive to researchers. Modern technology has made it possible for medicine to collect a vast amount of physiological data from patients during their course of treatment. Prior research has focused primarily on using this data for diagnosis, based on prior clinical evidence. But development of analytical tools for predicting medical conditions using real-time physiological data has been under explored. Early detection of medical complications and adverse conditions allow physicians to apply appropriate interventions that prevent

adverse outcomes. Even though predictive and preventative medical interventions are preferred over the reactive methods, the predictive medical analytics has not found its rightful place in the gold standard of medical practice. Further research and development are necessary to advance predictive medical analytics into robust, practical and every day standard medical tools.

Medical science is grounded in scientific evidence, prior research, experiments and studies that have produced a body of medical knowledge based on generalizations and meta-analysis of research data. Such generalizations explain the causal relationships between risk factors, diseases and diagnosis. There are however gray areas in medical prognostics because many health treatment and screening decisions have no single 'best' choice and because there is scientific uncertainty or the clinical evidence is insufficient (O'Connor, Bennett, Stacey, et al. 2009). In many areas of medical science, the causal relationships are still incompletely understood and controversial. There are environmental, situational, cultural and unique factors that provide specific clinical data about a disease or groups of patients. Although this data is inadequate for making scientific generalizations and clinical evidence, it can provide valuable information to make assessments of individual's health status.

Therefore, a computational model that can adapt to specific domains, patient demographics and geographies is desirable and useful in providing clinical predictions using available physiological data from patients under treatment. Since Artificial Neural Networks are able to model non-linear data relationships and to adapt to new data sets, they are promising tools for this computation model.

Advances in vital-signs monitoring software/hardware, sensor technology, miniaturization, wireless technology and storage allow recording and analysis of large physiological data in a timely fashion (Yu, Liu, McKenna, et al. 2006). This provides both a challenge and an opportunity. The challenge is that the medical decision maker must sift through vast amount of data to make the appropriate care decision. The opportunity is to analyze this large amount of data in real time to provide forecasts about the health of the patient and assist with clinical decisions.

However, clinical prediction models such as the model developed in this dissertation become more feasible for physicians to use for data mining and extracting relevant information to predict where the patient's health is headed.

A large volume of literature concerning mathematical models to predict biological and medical conditions has been published. But only a few of such works in predictive mathematical tools have found their way into mainstream clinical applications and medical practice. Several reasons are cited for the low adoption of predictive tools: either important biological processes were unrecognized or crucial parameters were not known, or that the mathematical intricacies of predictive models were not understood (Swierniak, Kimmel, Smieja 2009). When such parametric or evidence-based knowledge are not available, prognostics framework described in this research can be crucial to making clinical decisions.

Generally, most of the predictive methods previously proposed are based on a single model. The concept of ensemble of models (also known as committee of models) has received considerable attention in recent years. The main idea of ensemble of models is to combine the outputs of several models into a single predictor. While the concept of committee models has been used in other domains of research in the past, only a few, Ghavami and Kapur (2011) and this research have investigated the viability of committee models in the clinical domain, trained on clinical data for medical prediction.

1.3 Contributions

Recent advances in clinical data collection have given rise to the big data problem. This problem is visible in 3 dimensions of volume (large volume of data being collected and stored), velocity (the rate of collecting data is rising rapidly) and variety (there are a wide range of clinical data types and formats). Despite the rise in volume, velocity and variety of clinical data, little attention has been paid to developing a framework for prediction of patient health status using prognostic methods on these large data sets. This research develops and explores a multi-model prognostics framework for prediction and demonstrates its feasibility as a systemic prognostics method to predict patient health status. Following are three areas of contributions in this dissertation:

- First Contribution: It presents a control system model for Prognostics and provides a control theoretic feed-forward model for prediction of disease and medical intervention;

- Second Contribution: It introduces a prognostics framework using a committee (ensemble) of ANN models. This framework consists of three new ways of constructing models using clinical data for model training:
    a. It employs a multi-model ANN prediction framework and combines the results of each trained model to provide better validation and accuracy.
    b. It reviews various methods to compare accuracy of predictive ANN models and offers a framework for comparing accuracy,
    c. It combines ANN models with an oracle program to select more accurate predictions among a number of ensemble methods.
- Third Contribution: It studies the viability of using ANN models as predictive models to clinical data;

The first contribution of this dissertation is the control system treatment of prognostics and feed-forward model of clinical prediction. The other contribution is the evaluation of feasibility and viability of using ANN as a medical prognostics tool. Among feasibility elements, accuracy and generalizability of ANN models are examined. The second contribution is the comparison of multiple ANN algorithms using the same case study. The intent of using multiple ANN models is to account for diversity in clinical data types. Some ANN models are more accurate and perform better on certain class of data than others. No single algorithm can be ideal for different types or volume of clinical data. Using four independent ANN models ensures that results from a sampling of various models are considered. Finally, a framework for comparison of the four models on accuracy characteristics is provided. This framework along with a supervisory oracle program that selects the most accurate algorithm or ensemble of algorithms enhances the model's accuracy. To that end, this study investigates the results for a specific clinical case study, the prediction of a medical condition called DVT (Deep Vein Embolism).

The aim of this study is to improve the physician's ability to make proactive and preventative medical interventions, namely decisions, to prevent future complications and derive knowledge necessary for evidence-based medicine in advance.

There are five critical factors that define applicability, viability and feasibility of using ANN models as a prognostics tool. These factors fashioned after Smye and Clayton's

work on mathematical models in medicine can be defined by the following criteria (Smye, Clayton 2002):

1. Accuracy: Accuracy of prediction
2. Well-posedness: Stability & immunity to small perturbations of input data
3. Utility: Applicability, practicality and usability in the medical workflow
4. Adaptability: Ability to handle new evidence, i.e. new data values and data types
5. Economy: Cost of computation and timeliness of prediction

To explore the above critical factors in real case study, a prognostics model using four different types of neural network algorithm are compared in this research. The prediction accuracy of each model is compared to the actual clinical outcomes from prior retrospective patient cases. Analysis about accuracy includes measurements such as calibration (agreement between predicted probability and observed outcome frequencies), discrimination (ability to distinguish between patients with and without the disease), sensitivity (true positive rate, or proportion of patients who are correctly diagnosed as having the disease), specificity (true negative rate, the proportion of healthy patients who are correctly diagnosed with negative result), likelihood ratio (LR, the likelihood that a given test result would be expected in a patient with the disease compared to the likelihood that the same result would be expected without the disease) and receiver operating characteristic (ROC, a plot of true positive rate vs. false positive rate, namely a plot of sensitivity vs. one minus the specificity) curves (CEBM 2012).

Model validity (do the model's results match the reality), accuracy (the closeness of results to the quantity's actual value) and precision (the degree in which repeated measurements under unchanged conditions produce the same result) are compared among the four ANN models using clinical outcomes of prior patient data obtained from retrospective studies. But, clinical field validation and impact analysis on prospective cases are not considered in this research and are proposed as future research topics.

Most medical predictive models provide crude and general predictions based on risk factors over a long time horizon that range from several months to several years in the future. A major contribution of this dissertation is the development of a neural network predictive model with a trainable rule-based engine that provides continuous predictions of patient health status using real time patient physiological data. The goal of the prognostic model is

to offer predictions for a patient's health status over a short term time horizon. In other words predict adverse or abnormal conditions in a time period ranging from a few seconds to several hours from any given the current time.

The case study involves prediction of Deep Vein Thrombosis/Pulmonary Embolism (DVT/PE) complications. DVT/PE, a condition that causes blockage of patient lung vessels via blood clots leading to severe pain, loss of lung function and even death. The process and results of using an ANN model to predict (DVT/PE) are discussed.

1.4 Significance of This Research

The Evidence-based Medicine Working Group (McGinn, Guyatt, Wyer, Naylor, Stiell, et al. 2000) has proposed that prediction rules can "change clinical behavior and reduce unnecessary costs while maintaining quality of care and patient satisfaction."

Physicians struggle to incorporate the best evidence into their daily work. However, the vastness of clinical data and diverse patient situations make it difficult for physicians to remember all applicable CPRs and compute them correctly at any given moment. Developing a prognostics model that can learn and adapt to specific clinical environments and patient population is highly desirable.

Feasibility study of predictive models based on artificial neural networks to assist physicians as alternate scoring method to CPRs is a worthwhile research because it provides deeper clinical intelligence about the patients' clinical status and diagnosis. CPRs are a branch of evidence-based medicine. Examples include Wells score[1], Ottawa ankle rules[2], Ranson criteria[3] and Apache II[4]. It has been posited (Laupacis, Sekar, Stiell 1997) that the purpose of prediction rules is to "suggest a diagnostic or therapeutic course of action." Almost all prediction rules in use today provide diagnostic or prognostic probabilities, typically by using a score or risk-stratification algorithm.

Prognostics methods using Artificial Neural Networks (ANN) promise to deliver new insights into patient health status that provide more effective medical treatment during the

[1] Wells score is a clinical prediction rule developed by Wells that predicts a patient's probability of acquiring pulmonary embolism.
[2] Ottawa ankle rules are guidelines that doctors in deciding if a patient who presents with foot or ankle pain should be offered X-rays to diagnose a possible bone fracture.
[3] Developed in 1974, Ranson rule is a clinical prediction rule for predicting acute pancreatitis.
[4] APACHE II (Acute Physiology and Chronic Health Evaluation II) is one of several ICU scoring systems that offer severity of disease classification score, an integer between 0 to 71 to patients admitted to Intensive Care Unit (ICU).

patient hospital stay. Using predictive models as a standard of care promises to improve patient care and diagnostic quality by providing advanced knowledge of patients' impending health status. The application of Prognostics Health Management (PHM) to human physiological measurement data promises to conceptually deliver several benefits such as:

1) By continuously and iteratively assessing the physiological and biological input data provide prediction and advanced warning of medical complications.
2) A localized prediction tool that incorporates nuances of local population and clinical environment.
3) Ability to analyze large data sets and provide trained models for prediction.

Continuous and periodic monitoring of the individual's physiological systems involves collecting data from historical patient physiological data ranging from circulatory to respiratory and immune systems. The PHM method used in this research considers prognostic models built upon prior data from retrospective cases in order to make predictions about a new patient.

The following section reviews the prior research and published literature in these areas.

1.5 Prior Research & Literature Review

In order to cover relevant prior research literature critical to this study, studies from a confluence of three areas are examined. These areas are: Clinical Prediction Rules, Prognostics and Health Management, and an overview of clinical statistical methods including logistic regression, decision trees, Case-based reasoning (CBR). In addition, relevant prior research in Prognostics, Artificial Neural Networks and model validation are covered. A summary of prior research limitations are also presented. The diagram below is the epistemology of CPR research and literature review:

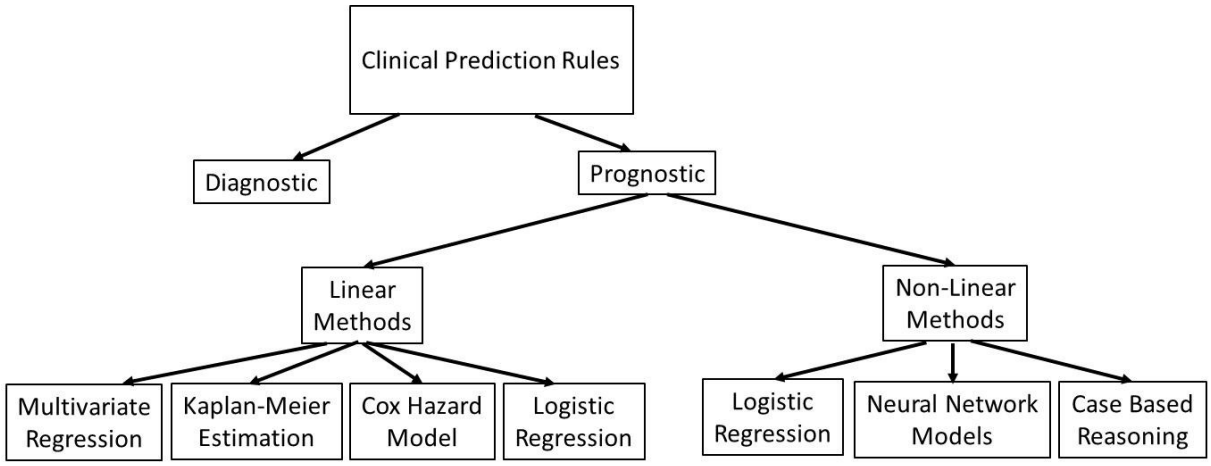


Figure 1: Epistemology of prior Clinical Prediction Rule methods

Modern healthcare strives to improve patient care and extend lifetime by using state of the art techniques and equipment. Among techniques available, predictive and preventive approaches are highly desired since they can reduce medical complications and reduce healthcare costs. As the sophistication and availability of medical devices and sensors have risen in recent years, more data is being collected that can aid in patient prognostics. Similarly, sensors and data acquisition components have become smaller and cheaper, thus their use in products has increased dramatically. This trend provides substantial, real time and often continuous flow of information about all aspects of a system. Availability of continuous sensory data improves the ability to diagnose and self-adjust systems for longer and more effective functional life. The same principles apply to patient health monitoring and prognostics. In the next sections, an overview of research and literature about medical prediction methods are summarized.

1.5.1 Prior research in Medical Prediction Methods

The American Heritage Dictionary defines *prognostics* as an adjective that relates to prediction or foretelling and as a noun for a sign or symptom indicating the future course of a disease or sign or forecast of some future occurrence. Hippocrates founded the 21 axioms of prognostics some 2400 years ago (MIT 2010). The goal of prognostics is to foretell (predict) the future health (or state) of a system. Health is defined as a state of complete physical, mental, and social well-being.

There are four philosophies pertaining to biological and medical prediction that have evolved through the history of medical prediction. One is grounded in control theory. Decay

of human physiology and adverse medical conditions such as Intra-Cranial Pressure (ICP) or carcinogenesis can be viewed as a result of loss of body's control over it critical mechanisms. For example, loss of control over blood flow regulation leads to irregular intracranial pressure; or loss of control over cell cycle that causes altered function of a certain cell population leading to cancer. Medical intervention is viewed as a control action for which the human body is the system. This approach requires a deep understanding of the internal causal models between control mechanisms and human physiology. An example of this approach are the mathematical models using control theory that have employed differential equations to synchronize administration and dosing of chemotherapy drugs with optimum timing of cell life cycle.

The second approach follows the Markov chain model as it considers the disease cycle as a sequence of phases traversed by each physiological subsystem from birth to expiration. For example, a patient with pneumonia starts from healthy, normal state and then follows four stages of Congestion, Red hepatization, Gray hepatization, Resolution (recovery). As another example, a cell cycle consists of growth, DNA synthesis, preparation for division) and division. More recently formulations that model cancer cell growth and decay cycles have been proposed (Hahnfeldt, Danigraphy, Folkman, et al. 1999). These models have considered both deterministic and probabilistic approaches.

The third type of mathematic construct considers the asynchronous nature of biology and uses simulation models. For example, one study applied simulation and statistical process control to estimate occurrence of hospital-acquired infections and to identify medical interventions to prevent transmission of such infections (Limaye, Mastrangelo, Zerr, et al. 2008). Other predictive models in cancer therapy have used stochastic process to predict drug resistance of cancer cells and variability in cell lifetimes. Such a stochastic process is a random walk superimposed on the time-continuous branching process of cell proliferation, namely a branching random walk (Kimmel, Axelrod 2002).

The fourth approach considers the human body as a "gray-box". Since perfect knowledge about each individual's physiology, environmental, genetic and cultural information is not available and in the areas of medicine where our knowledge of clinical evidence is uncertain, one can only rely on predictive models that take data from physiological sensors and laboratory results to make predictions.

Some of the models in this category include the survival analysis provided by Cox Hazard model and Kaplan-Meier estimate. The term survival analysis comes from biomedical research in study of mortality, or patients' survival times from the time of diagnosis of a disease to death. The first survival analysis was developed by John Graunt (Graunt 1662) who for the first time developed life tables based on his birth-death rate observations. Survival analysis is a collection of statistical techniques used to estimate whether an event of interest will occur and at what time. Survival Analysis is known by different names in different disciplines; engineering researchers refer to it as failure-time analysis; sociologists call it event history analysis while economists call it transition analysis.

Among parametric models of survival analysis the Cox hazard function is a popular method. Another method called Kaplan-Meier Estimator is a statistical tool used for non-parametric models where the mathematical equation of the system under study is unknown. Prior to these models, researchers often had to resort to life-table methods.

The Cox hazard model is a partial likelihood method that allows the researcher to estimate the regression coefficients of the proportional hazards model without the need to specify the baseline hazard function. The hazard rate is the probability that if the event in question has not already occurred, it will occur in the next time interval, divided by the length of that interval (Spruance, Reid, Grace, Samore 2004).

Among biomedical researchers the Kaplan Meier Estimator is the tool of choice for survival analysis. Also known as the product-limit estimator, this technique observes all data from all observations by considering survival to any point in time as a series of steps defined by the observed survival and censored times. Some models use Taylor series expansion. It's often used to measure the fraction of patients living for a certain period of time after treatment. One advantage of Kaplan Meier is in its ability to handle censored data, those situations for example where a patient withdraws from a study or the certain start time of the data is not available. Kaplan Meier makes certain assumptions about data independence and uniformity that if violated can result in biased and unreliable data (Tsai, Pollock, Brownie 1999).

Other researchers have adopted a Case Base Reasoning (CBR) method for diagnosis. CBR is an approach for solving a new problem by remembering a previous similar situation and by reusing information and knowledge of that information (Aamodt, Plaza 1994). Since

this approach assumes that similar problems have similar solutions, it is considered an appropriate method for practical medical domain that's focused on real cases rather than rules on knowledge to solve problems (Park, Kim, Chun 2008).

The emphasis of this study is on prediction of the individual's short term future health condition and a rule-based prognostics engine that makes such predictions possible. Short term is defined as a time frame that spans from a few seconds to several days from any given moment. The prognostics engine is a computational component that can analyze vast amounts of historical and current physiological data and predict future health of an individual. The predictions are continuous over time based on new, real time data gathered from multiple physiological systems including warnings, alerts, events and precautions.
Admittedly developing mathematical models that make accurate predictions in biology and medicine is challenging but researchers suggest that soon such mathematical models will become a useful adjunct to laboratory experiment (and even clinical trials), and the provision of 'in silico' models will become routine.

Advances in vital-signs monitoring software/hardware, sensor technology, miniaturization, wireless technology and storage allow recording and analysis of large physiological data in a timely fashion (Yu, Liu, McKenna, et al. 2006). This provides both a challenge and an opportunity. The challenge is that the medical decision maker must sift through vast amount of data to make the appropriate care decision. The opportunity is to analyze this large amount of data in real time to provide forecasts about the health of the patient and assist with clinical decisions.

Medical science is grounded in scientific evidence, prior research, experiments and studies that have produced a body of medical knowledge based on generalizations and meta-analysis of research data. Such generalizations explain the causal relationships between risk factors, diseases and diagnosis. There are however gray areas in medical prognostics because many health treatment and screening decisions have no single 'best' choice and because there is scientific uncertainty or the clinical evidence is insufficient (O'Connor, Bennett, Stacey, et al. 2009). In many areas of medical science, the causal relationships are still incompletely understood and controversial. There are environmental, situational, cultural and unique factors that provide specific clinical data about a disease or groups of patients.

Although this data is inadequate for making scientific generalizations and clinical evidence, it can provide valuable information to make assessments of individual's health status.

There are situations where collected data is inadequate to make generalizations, the evidence is not full proof or inconclusive, and the collected data requires physician processing and judgment, these fall into a gray area of medical diagnosis and prognosis. This research presents a model based on ANN algorithms as alternative to linear logistics regression methods to discern patterns and classifications of collected data into specific disease or stages of disease classifications.

A large volume of literature concerning mathematical models to predict biological and medical conditions has been published. But only a few of such works in predictive mathematical tools have found their way into mainstream clinical applications and medical practice. Several reasons are cited for the low adoption of predictive tools: either important biological processes were unrecognized or crucial parameters were not known, or that the mathematical intricacies of predictive models were not understood (Swierniak, Kimmel, Smieja 2009). When such parametric or evidence-based knowledge are not available, predictive model described in this research can be crucial to make clinical decisions.

The properties of an appropriate mathematical model for medical health condition include: accuracy, prediction, economy, well-posedness and utility (Smye, Clayton 2002). Among constructs used in prior research several distinct mathematical models can be found, such as: Multivariate regression analysis, Markov chains, stochastic processes, Bayesian networks, Fuzzy logic, control theory, discrete event simulation, dynamic programming and Neural Networks.

While control theory has been widely used in systems and other engineering disciplines, it has not widely been applied to prognostics as explained in the next section.

1.5.2 Control Theoretic Approach to Prognostics

A system can be broadly defined as an integrated set of elements that accomplish a defined objective (INCOSE 2008). The human body can be considered as a biological system that functions as a collection of interrelated systems. One of the questions that this research intends to answer is how Prognostics and Health Management methodology may be applied to human biological systems as a methodology to predict and prevent adverse medical conditions in patients.

Generally, a control theoretic approach to PHM has not been adequately addressed in the literature to the degree that this research does. In particular the system theory approach in this research that covers both feed-forward and feedback mechanisms delivers a richer framework for Prognostics.

Prognostics deals with prediction of some desired quality or characteristic of a system (Kapur 2010). It's based on understanding the science of degradation of the underlying system. The traditional systems control has been predominantly based on feedback: Using the feedback signal, a system's performance could be diagnosed, then adjusted to fix the problem. Obviously, this poses an issue. Correcting the problems after receiving feedback might be too late in certain mission critical systems and in particular when human physiology is considered. Instead, a feed-forward model as exemplified by Prognostics methods would be more desirable. Prognostics methodology is based on two principles: 1) it uses feed-forward models for prediction or forecast of the underlying causes of problems by analyzing feed-forward signals; and 2) it suggests changes in input signal to the system in order to prevent the problem from occurring.

Prognostics and Health Management (PHM) is an engineering discipline that links studies of failure mechanisms to system lifecycle management (Serdar, Goebel, and Lucas, 2008). Other definitions of PHM describe it as a method that permits the assessment of the reliability of a system under its actual application conditions, to determine the advent of failure, and mitigate system risks (Pecht, 2008).

The term "diagnostics" pertains to the detection and isolation of faults or failures. "Prognostics" is the process of predicting a future state (of reliability) based on current and historic conditions (Vichare and Pecht 2006).

Prognostics is the science of predicting the future functionality of a system by estimating the remaining useful life, probability of failure or time to failure for a given system. There are many approaches and modeling frameworks for representing a prognostics system.

Among many methods of prognostics computing the Remaining Useful Life (RUL) of a system is common. RUL can be estimated from historical and operational data collected from a system. Various methods are used to determine system degradation and predict RUL.

Another method is estimating the Probability of Failure (POF). POF is the failure probability distribution of the system or a component. Additionally, it's common to study Time to failure (TTF), the time a component is expected to fail. TTF defines the time when a system no longer meets its design specifications.

Prognostics and Reliability are interrelated. Reliability is defined as the probability that product will perform its intended function, satisfactorily for its intended life when operating under specified condition (Kapur 2010). A clinical definition can be derived from this technical description; Reliability is the probability that a patient will not develop certain medical complication during the length stay under medical care of the care provider(s). Reliability is measured by several indicators such as Mean Time Between Failure (MTBF), Failure rate and Percentiles of Life. Each measurement can be computed from corresponding equations that are derived from empirical and statistical distribution functions. Additional treatment of Reliability can be found from text by Kapur and Lamberson (Kapur, Lamberson 1977).

Prognostics models can be classified into three general types (Eklund 2009; Hines 2009; Peysson et al. 2009). Type I is reliability based. It applies the traditional time to failure analysis by tracking a population of failures and using statistical methods for the estimation of reliability. Some typical life distributions that are used in this type of prognostics include Weibull, exponential and normal distributions. Type I prognostic methods does not incorporate the real time monitoring of operating conditions or environmental conditions.

Type II methods, also known as the stressor-based approaches consider the operational and environmental condition data. Type II considers the failures of a system in its operating environment to provide an average remaining life of a component. Some of the environmental data might include temperature, vibration, humidity and load. The proportional hazard model is an example of a type II prognostic model. Knowing the causes, one can predict reliability of a system. The simplest model in this approach is the regression model: given the operating and environmental conditions, one can predict the system failure and remaining useful life by a regression equation.

Type III prognostic methods are condition-based, namely they characterize the lifetime of a system in operation within its specific environment. They estimate the

remaining life of a specific component or the entire system. Among methods used in Type III prognostics are the General Path Model (GPM), Neural Network models, Expert systems, Fuzzy rule-based systems, and multi-state analysis. Another example of type III approach is the cumulative Damage model.

The cumulative damage model tracks the irreversible accumulation of damage in systems or components. The statistical cumulative damage model considers the number of possible damage states and a transition matrix (for representing a multi-state Markov Chain) to provide a damage prediction for multiple cyclical loads.

It's the Type III Prognostics method using Artificial Neural Network models that this dissertation employs to continuously predict a patient's health status at regular time intervals.

1.5.3 Artificial Neural Networks in Predictive Medicine

Artificial Neural networks (ANNs) are parallel computational methods by interconnecting artificial neurons. They're ideal for solving non-linear problems that come with a long list and diverse types of input variables. ANNs are adaptive to specific problems and can be trained for pattern matching or classification. An ANN model can be trained by mapping a disease to a known set of input clinical measurements and then later be applied to a new patient. The trained model can match the input measurements of the patient to presence or absence of a disease. The model can even classify the patient's clinical measurements into various stages of a disease.

Since early 1980's, Artificial Neural networks have been applied successfully to several prediction problems in business, engineering and medicine (Delen 2009). One of the most popular models is the Multi-Layer Perceptron (MLP) with back propagation, essentially a supervised learning algorithm. It's been shown that ANN models using MLP algorithm are capable of learning arbitrarily complex non-linear functions to arbitrary accurate levels (Hornik 1990). The MLP is essentially a collection of non-linear neurons connected together by weighted links in a feed-forward multi-layer structure. Among highly accurate models, Support Vector Machines (SVMs) have been proposed (Delen 2009). Most recent algorithms use Levenberg-marquardt methods proposed by Levenberg and Marquardt that are highly accurate as well as computationally fast ANN models (Wilamowski & Chen, 1999).

Neural Networks have been used successfully to predict future onset of diseases such as recurrence of various types of cancer, cardiology illnesses and to assist physicians with prognostic and decision support. These studies have offered long term predictions for patient health conditions, typically forecasting the disease-free or disease recurrence in the future ranging from a few months to several years.

Medical research has shown that certain life-threatening conditions exhibit early indicators in physiological data. A study conducted on improving neonatal intensive care units (NICU) (Blount 2010) provided interpretations of multiple streams of clinical and physiological data to detect medically significant conditions that precede the onset of medical complications for neonatal patients.

In another study (Webber 1994), A neural network model was trained on EEG data. The input consisted of 49 channels of real time EEG data to detect epilepsy spikes. The study showed that ANNs offer a practical solution for automated detection of real time epileptiform discharges using inexpensive computers.

ANNs have been shown to be a valuable tool to the clinical diagnosis of myocardial infarction (Baxt 1994). The model used in one study was trained on 351 patients admitted for high likelihood of having myocardial infarction. It was prospectively tested on 331 consecutive patients presenting to the ED department with anterior chest pain. The network was able to distinguish patients with from those without acute myocardial infarction at a slightly higher sensitivity than physicians' diagnosis for those patients.

In another study of patients in Intensive Care Units (ICUs), an ANN model was shown to be more effective than logistic regression model for predicting outcome of care (Dybowski, et al. 1996). The ANN model was applied in the clinical setting of systemic inflammatory response syndrome and hemodynamic shock on 258 patients. The outcome evaluated was death during that hospital admission. The best performing ANN model was trained after 7 training iterations.

In cancer treatment cases, ANNs have become a popular tool for predicting outcomes (Dayhoff, DeLeo 2001). At one institution (Bottaci, et al. 1997) six different ANN models were developed to predict outcome of individual patients who were diagnosed with colorectal cancer to predict death within 9, 12, 15, 18, 21, and 24 months. Results showed that ANNs

were able to detect outcome more accurately than the then available clinicopathological methods.

Other research conducted in the breast cancer patients (Ravdin, Clark 2005) suggest that ANNs can be trained to recognize patients with high and low risk of recurrent disease and death. Moreover, their study showed that by coding time as one of the prognostic variables, an ANN can be used to predict patient outcome over time. In particular ANN models can make a series of predictions about probability of relapse at different times of follow up, allowing clinicians to draw survival probability curves for individual patients.

In another study a set of patients' mammography tests were interpreted by radiologists and by an ANN model (Floyd, et al. 1994). The model was more accurate in detecting breast cancer patients than radiologists. A more comprehensive overview of application of neural networks in decision support of cancer found 396 studies and found that overall ANNs add more benefit to making decisions in the field of cancer (Lisboa et al. 2005).

Several studies have compared the accuracy of ANNs with Logistic regression models, but after a meta-analysis the conclusions are mixed. Some papers (Delen 2009) show that ANNs are far more accurate, but a few papers find both methods comparable for medical prediction (Adams, Wert 2005). The next section reviews the validation and viability measurements that are used as criteria to select the most appropriate model.

1.5.4 Model Viability and Validation Methods

Model validation and verification are important steps for providing confidence and credibility in the model's results. Verification (ensures that the model performs as intended) and validation (ensures that the model represents and correctly reproduces the behaviors of the real world system) are essential elements of model development for practical applications (Macal 2005). Model verification deals with building the model right. Validations deals with building the right model.

The goal of validation is to ensure that the model addressed the right problem, provided accurate information about the system being modeled. One of the dangers to modeling validity is "overfitting" the model to a given data set, where the model is fitted to a specific dataset. Overfitting can occur when important elements of the model reflect randomness in the data rather than underlying model drivers. To overcome this limitation,

researchers have employed techniques such as cross-validation, or keeping a "hold-out" random data sample to perform testing on a separate data set. In addition, researchers have considered accuracy measures of the model using prior data as the expected results.

Three aspects of validity are advised:

i) Calibration - agreement between observed probabilities and predicted probabilities,

ii) Discrimination - ability of the model to distinguish between different outcomes,

iii) Clinical usefulness - ability of the model to improve decision making process.

There are two types of validation, the internal validation and external validation. Internal validation uses techniques such as cross validation and boot-strapping to assess the performance in samples of the same population. External validation is the process of measuring performance of prediction model in samples from different populations such as patients from other locations. With cross-validation technique, the model is developed in a randomly selected part of the data sample and test on the rest of the sample, then the process is repeated several times and the average is computed as the estimate of performance. With boot-strapping technique, a sample of the same size as the development sample is randomly selected with replacement, the model then is developed in the boot-strap samples and testing on those not included in the boot-strap samples.

Tests that measure clinical usefulness include accuracy, sensitivity, specificity and decrease in weighed false classifications. Tests that measure discrimination include Receiver Operating Characteristic (ROC) curve, a plot of model sensitivity vs. (1 – specificity). Finally tests that measure calibration include calibration plot and average absolute different between observed frequencies and predicted probabilities. These measurements are defined and employed in this research as described in the upcoming chapters.

External validation requires clinical trials and specific clinical design of experiments that are outside the scope of this research. However, internal validity tests will be performed on all four ANN models and the five derived ensemble of models.

1.5.5 Limitations of prior research in medical prognostics

A vast majority of mathematical models in medicine are used in diagnosis. Models to make prediction using prognostics have not been fully explored. Among those that did address predictive analytics, many required and made assumptions about the type of data and distribution. For example, many studies assumed normality in their data set. Most have relied on a single model to make predictions. Little attention has been paid to developing a framework for prediction of patient health status using prognostic methods on these large data sets. This research develops and explores a multi-model prognostics framework for prediction and demonstrates its feasibility as a systemic prognostics method to predict patient health status.

Traditionally, the two most commonly used data mining techniques are Linear Discriminant Analysis (LDA) and logistic regression to construct classification models. However, one criticism leveled against LDA has been due to assumption about the categorical nature of the data and the fact that the covariance matrices of different classes are unlikely to be equal. Research in cancer data analysis has demonstrated that generally ANNs are more accurate than linear logistic regression models in predicting cases of new or recurrent cancer (Delen 2009).

From a survey of literature from 1970s to present, it's established that more attention has been given to decision support and diagnoses and less to predicting short term medical health condition of individual patients. The most successful predictive methods in literature are model-free approaches using neural networks and fuzzy sets (Kodell, Pearce, Baek, et al. 2009, and Arthi, Tamilarasi 2008).

CPRs are typically arrived at through logistic regression type of analysis. Logistic regression is a special form of linear regression models that allows non-numeric input variables. It's used for prediction of the probability that an event will occur by fitting data to a logit function (or a natural log function) logistic curve.

Logistic Regression (LR) is a generalization of linear regression. It's used as the means to predict binary or multi-class dependent variables. Since the response variable is discrete, it cannot be modeled directly by linear regression. Instead, logistic regression rather than predicting a point estimate of the event itself, builds the model to predict the odds of its occurrence. When predicting the occurrence or no-occurrence of a disease, basically a two-

class problem, if the odds are greater than 50%, it implies that the case is assigned to the class designated as 1, otherwise as 0. The LR assumes that the response variable (the log of odds) is linear in the coefficients of the predictor variables. In addition, the LR models do not select the best inputs and the modeler must select the right inputs and specify their relationship to the response variable (Delen 2009). These are among limitations to prior research that have employed Logistic Regression.

Classification trees (also referred to as decision trees) are used to predict membership of cases in the classes of categorical dependent variable based on their measurements on predictor variables. Researchers prefer classification trees over the traditional statistical tools when assumptions about data distribution cannot be met or the researcher seeks exploratory study of data classification. Classification trees have been used in medical studies (Breiman, Friedman, Olshen, Stone 1984) to aid in diagnosis. They form hierarchical models of data with branches in a tree-like structure that lead to specific diagnosis. The pitfalls with these methods are shown to be related to their linear approach and errors associated with initial choice of data that forms the tree classification.

Historically, most predictive methods have relied on either a single model for prediction and/or on linear methods for prediction and generalizing CPR rules. While predictive models using ANN have been reported in the literature, such models provide long term predictions that span over several years in the future, and do not focus on short term predictions.

Prior research has predominantly limited their model validation to a few criteria, such as sensitivity, specificity and ROC calculations. They have limited their models to only one ANN model or to the algorithm that produced the best results. They have not compared results of four ANN models along these and other validation measurements which this research has addressed.

Given these limitations, this research proposes a multi-model prognostics framework along with an oracle to select the most appropriate model for a given disease prediction. Furthermore, this research compares a wide range of validity measures on all four models and ensemble of these models. The use of multi-model prediction and in particular using ensemble of models combined with an oracle overseer as explored in this research are novel approaches in clinical data analytics.

Chapter 2

2 PROPOSED METHODOLOGY OVERVIEW

The prognostics model presented in this research provides a feed-forward model to make predictions based on models trained on prior patient data. A prediction is a form of speculation about a state in the future. A prediction is foretelling a medical event or disease when the ingredients for that medical event are in place but have not combined to affect their significance in form of a disease yet. A marker is the recognition that the ingredients for a medical event are in place and have indeed combined to result in form of a disease but in lower and milder yet measurable doses.

The precursor to a disease is known as risk factors in medicine. Thus, the spectrum of medical predictions starts with risk factors, leading to prediction, and then on to markers and finally to the occurrence of the disease or medical event itself. The following graph illustrates the chronology of events and progression of the individual's health status from risk factors towards confirmed stage of disease or medical event manifestation. The distance between time ticks are arbitrary and vary among individuals. The medical prediction models must take into account the prior history, risk factors, markers and the medical intervention as inputs to the model. Once medical intervention is applied, patients' physiological data is expected to reflect recovery and return of patient health condition into the desired range. Such recovery can be expected and compared against measured changes in the patients' health status. The intent of this research is to study the viability of using ANN to make predictions in the time scale shown in Figure 2.

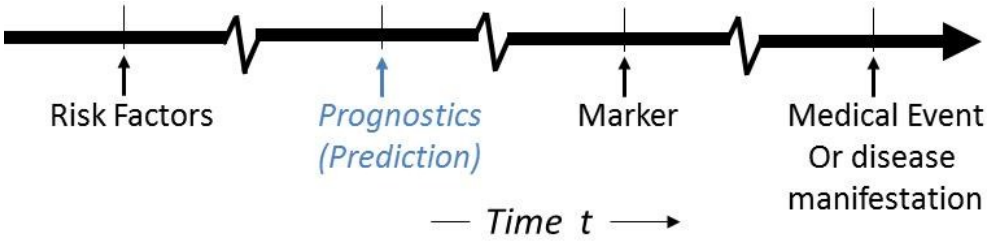


Figure 2. Progression of individual health condition

2.1 An Overview of Predictive Mathematical Models

According to Smye and Clayton, the properties of an appropriate mathematical model for medical health condition include: accuracy, prediction, economy, well-posedness and utility (Smye, Clayton 2002). Among constructs used in prior research several distinct

mathematical models can be found, such as: Multivariate regression analysis, Markov chains, stochastic processes, Bayesian networks, Fuzzy logic, control theory, discrete event simulation, dynamic programming and Neural Networks.

There are five evolving philosophies pertaining to biological and medical prediction: One is grounded in control theory. Decay of human physiology and adverse medical conditions such as Intra-Cranial Pressure (ICP) or carcinogenesis can be viewed as a result of loss of body's control over it critical mechanisms. For example, loss of control over blood flow regulation leads to irregular intracranial pressure; or loss of control over cell cycle that causes altered function of a certain cell population leading to cancer. Medical intervention is viewed as a control action on the specific physiological subsystem, for which the human body is the general system. This approach requires a deep understanding of the internal causal models between control mechanisms and human physiology. The mathematical models using control theory have employed differential equations to synchronize administration and dosing of chemotherapy drugs with optimum timing of cell life cycle. Such models have offered treatment protocols that maximize the effect of drug dosages on cell subpopulation while minimizing impact on healthy cells.

The second approach follows the Markov chain model as it considers the disease cycle as a sequence of phases traversed by each physiological subsystem from birth to expiration. For example, a patient with pneumonia starts from healthy, normal state and then follows four stages of Congestion, Red hepatization, Gray hepatization, Resolution (recovery). As another example, a cell cycle consists of G1 (or growth), S (for DNA Synthesis), G2(preparation for division) and M(division). A similar model in this category is the Gompetz Curve, a sigmoidal function which has been used to predict cancer cell growth. More recently formulations that model cancer cell growth and decay cycles have been proposed (Hahnfeldt, Danigraphy, Folkman, et al. 1999). These models have considered both deterministic and probabilistic approaches.

The third type of mathematic construct considers the asynchronous nature of biology and thus this approach uses simulation models. For example, one study applied simulation and statistical process control to estimate occurrence of hospital-acquired infections and to identify medical interventions to prevent transmission of such infections (Limaye, Mastrangelo, Zerr, et al. 2008).

The fourth approach such as those used as predictive models in cancer therapy have used stochastic process to predict drug resistance of cancer cells and variability in cell lifetimes. Such a stochastic process is a random walk superimposed on the time-continuous branching process of cell proliferation, namely a branching random walk (Kimmel, Axelrod 2002).

The fifth approach considers the human body as a black box. Since perfect knowledge about each individual's physiology, environmental, genetic and cultural information are not known and in the areas of medicine where our knowledge of clinical evidence is uncertain, one must rely on predictive models that take physiological sensor data and laboratory results to make predictions.

2.2 Conceptual Model

This research offers a feed-forward model based on Prognostics and Health Management methodology. The model consists of several key components: Input, output, measured data, database of prior cases, a prognostics engine and the feed-forward signal as shown in Figure 3. The rule-based engine, essentially a classification tool uses ANN models to make prediction.

The model proposed by this research considers the medical treatment plan as an input to the patients' physiological system. Represented by *u(t)*, medical treatment plan involves some set of medications, procedures and care protocols prescribed by the physician. The patients' physiology is the process that produces a clinical outcome at time *t*, shown by *y(t)*. The patients' clinical outcome is the output or the response variable. The outcome is a vector of single or multiple states of health for that patient. The model is shown in Figure 3.

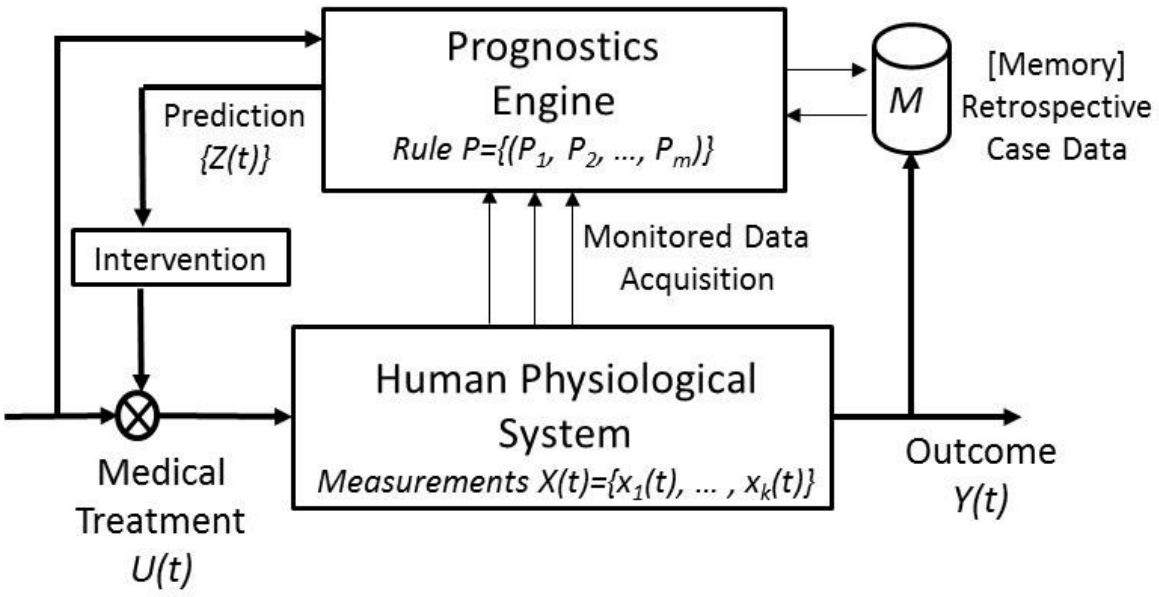


Figure 3. The Medical Prognostics Model

The input variable can be shown as:

$$U(t) = \{u_1(t),\ u_2(t), \ldots, u_q(t)\} \tag{1}$$

The output, known as the clinical outcome represents the patient's health status. This is represented by the response variable and defined as:

$$Y(t) = \{y_1(t),\ y_2(t), \ldots, y_m(t)\} \tag{2}$$

The input vector *U(t)* indicates which of medical protocols, procedures and treatments are applied at time *t*. Each $u_i(t)$ indicates a unique treatment defined by CPT codes for time *t*. The complete list of procedures is codified and can be found in a library of Current Procedural Terminology (CPT) code which includes approximately 8,700 unique procedures (American Medical Association 2010).

Vector *Y(t)* is the response variable, representing a set of diseases at time *t*. The diseases correspond to ICD-9 diagnosis codes found in ICD-9 library (WHO 2012). A response variable $Y_i(t)$ indicates the value or stage of disease *i* at time *t*. The value can be binary (1 or 0, namely True or False, indicating the presence or absence of a disease), or a discreet value (indicating the stage or class of a disease). For example, let's select a variable from the set *Y(t)*, say variable $y_{18}(t)$, to denote the presence of lupus. Lupus has five stages.

When the value of $Y_{18}(t)$ is 3, it indicates that lupus is present and it's in stage 3. But if the value of $Y_{18}(t)$ is 0, it implies the presence of lupus for the patient is negative.

The internal physiological system measurements consisting of clinical and vital sign data can cover a wide range of measurements including lab results, Radiology exams and real time information. These measurements are represented by the set $X(t)$:

$$X(t) = \{x_1(t), x_2(t), \dots, x_k(t)\} \tag{3}$$

The medical prognostics model employs a prognostics engine that consists of a set of pre-trained mathematical models to predict specific outcome for time $(t + t_1)$. Each mathematical model represents a dedicated rule for predicting a specific disease value. The rules are defined by the set P. In other words, P is the set of prediction rules, namely the set of trained ANN models that make the disease predictions. The prognostics engine works as follows: it collects vital clinical data from the patients' physiological system and makes a prediction for t_1 minutes in advance, for time $(t + t_1)$. The prognostics engine delivers a prediction vector $\{Z\}$ that can be used to modify the medical treatment plan $u(t)$.

Let's define clinical outcome $Y(t)$ as a function of medical intervention, physiological measurements and unknown cause-and-effect variables that can be accumulated into error $\hat{e}$:

$$Y(t) = F(\{X(t)\}, \{U(t)\}, \{Z(t)\}, \hat{e}) \tag{4}$$

The prediction rules are trained based on prior evidence and formed from retrospective collection of past patient data. The set of prognostic rules can be defined by:

$$P = \{p_1, p_2, \dots, p_m\} \tag{5}$$

The prediction rules are models defined by analyzing retrospective cases. Each prediction rule is defined by a model that transforms the input data set $X(t)$ to a particular disease Y_i. The prognostics engine works continuously by monitoring real time patient data and simultaneously applying multiple mathematical algorithms p_i every so many established minutes such that it can make predictions about occurrence of diseases or adverse events occurring in the near future. The prognostics vector $Z(t)$ consists of m values, indicating predictions about disease type and value for that disease shown by $z_i(t)$:

$$Z(t) = \{z_1(t), \dots, z_m(t)\} \tag{6}$$

The medical intervention, retrospective case information and monitored data can be mathematically described as sets of variables. One can express prediction as a function of multiple variables including the input clinical data and medical intervention. Prediction is a mapping between new input data and an outcome from a set of retrospective cases. The most suitable mapping is selected by a classification function defined by the set of rules p_i for disease i. The classification function maps a set of input variable data pattern with a specific disease using rules p_i. Each rule p_i is trained to detect disease i, so there is a one-to-one association between each rule p_i and disease i. Prediction for time $(t+t_1)$ is a vector of predicted disease values:

$$z_i(t + t_1) = p_i\{X(1), \dots, X(t), U(1), \dots, U(t), Y(1), \dots, Y(t)\} \quad (7)$$

where physiological data set collected from the patient is represented by vector $X(t)$; medical treatment plans are selected from a set of treatment plans shown as $U(t)$; and retrospective cases are represented by M as the set of prior relationships established between physiological data and outcome. The relationship between prediction and predicted disease at time $t+t_1$ can be shown as:

$$Prediction_{(t)} = Z(t), \xrightarrow{yields} \{Y_1(t + t_1), \dots, Y_i(t + t_1), \dots, Y_m(t + t_1)\} \quad (8)$$

The goal of this research is to identify the appropriate mathematical model $P(X, U, Y)$ that selects the appropriate prediction from a set of possible outcomes, in other words determine the value of $Y_i(t+t_1)$ as True, False or other discreet values indicating the stage of disease. The value of $Y_i(t+t_1)$ determines the classification for a given patient and answers the question of which disease classification the patient belongs to. Each p_i model is a mapping function developed based on historical data patterns that maps the input data to a specific outcome. The function $P(X, U, Y)$ is a classification function that selects one or more clinical disease states from the set Y.

The model is intended to be used by physicians. A typical use-case scenario is as follows: A physician, the user of this tool, will train the ANN model based on a-priori data and outcomes. When the ANN model is trained, the physician can apply the model to new patient data. When the ANN model is applied once, it provides diagnostic and prognostic classifications. The ANN model can be trained over a time interval taking into account a time

series of clinical data points. The model can be set up to run repetitively every few minutes for one or more patients to provide real-time and ongoing predictions about upcoming presence or absence of a clinical condition. The next section describes the four ANN models and their characteristics that are employed in this research.

2.3 Artificial Neural Network models

The power of artificial networks comes in its ability to detect patterns, including those complicated situations when the traditional statistical analysis would take an inordinate amount of time that would render them impractical (Monterola, Lim, et al 2002).

An artificial neural network is a network of interconnected processing elements that can classify patterns from a set of input data. Unlike the traditional computer architectures, known as von-Neumann computers, ANNs are trained, rather than programmed. When a set of data is fed into an ANN model if there is a pattern in the data, the ANN 'learns' them. Once the pattern is learned, the ANN model can classify a new set of data into the appropriate categories.

The suitable predictive mathematical model must offer accuracy and simplicity to learn from prior cases and easily be extensible to apply new data to make predictions about a patient's health condition. The four ANN algorithms selected in this research are established through literature among the most commonly used and accurate neural network models for prediction and classification. Each model has certain relative strength and weakness depending on the input data and computational constraints (Principe 2011). This research developed all four models for comparison. The models are:

1) PNN - Probabilistic Neural Networks are four layer networks. They classify data in a non-parametric method and are less sensitive to outlier data. These models are known for performing well when datasets are small.

2) SVM – Support Vector Machine networks. SVM performs classification by constructing a two-layer network that defines a hyperplane that separates data into multiple classifications. This method is generally regarded among more accurate classification models.

3) MLP trained with LM – Multi-layer perceptron with Levenberg-Marquardt algorithm, a gradient descent approach with variable step modification. This algorithm is regarded as computationally efficient method.

4) GFN (Generalized Feed-forward network) trained with LM – Generalized multi-layer feed-forward network with Levenberg-Marquardt algorithm. These models typically perform well when datasets are large and many data cases are available.

In order to make predictions on time-series data, a time-lag recurring network variation may be used for each of the above algorithms. The time-lag recurring network is essentially a time-series modeling approach that shifts the prediction several iterations forward in time and provides results of several samples ahead.

Different neural network models use different learning rules, but in general they all determine pattern statistics from a set of training examples and then classify new data according to the trained rules. Stated differently, a trained neural network model classifies (or maps) a set of input data to a specific disease from a set of diseases.

2.4 Model Evaluation

Feasibility and utility of the model is gauged against five criteria of accuracy, well-posedness, utility, adaptability and economy. Each criterion is explored further in the following sections.

2.4.1 Accuracy

Accuracy of a model is the degree of closeness of the model's results to the system's actual value. Precision of a model is the degree to which repeated runs of the model under unchanged conditions produces the same results. Since the model is trained on retrospective data, it's easy to evaluate the prediction accuracy of the model to the actual clinical outcomes from prior retrospective patient cases. Analysis about accuracy will include measurements such as calibration (agreement between predicted probability and observed outcome frequencies), discrimination (ability to distinguish between patients with and without the disease), sensitivity (proportion of patients who are correctly diagnosed as having the disease), specificity (proportion of healthy patients who are correctly diagnosed with negative result), likelihood ratio (LR, how much the odds of disease change based on a positive or negative test result) and receiver operating characteristic (ROC, a plot of sensitivity vs. one minus specificity) curves.

This study evaluates model validity and accuracy through internal validation using clinical outcomes of prior patient data obtained from retrospective studies. Clinical

validation and impact analysis on prospective cases are outside the scope of this research. This dissertation proposes an oracle schema to select the most accurate model, or to select an ensemble of models that provide higher accuracy of prediction.

Comparing prediction accuracy of ANN and other statistical models requires standards for comparison using classification performance indices. These indices include Receiver Operating Characteristic (ROC, a plot of sensitivity vs. one minus specificity) curve, Area Under Receiver-Operating Characteristics (AUROC, an overall measure of accuracy that measures the area under ROC curve and where bigger area indicates higher accuracy), sensitivity, specificity, accuracy and positive predictive value (PPV, probability that someone with a positive test result to actually have the disease) and negative predictive value (NPV, probability that someone with a negative test result to actually not have the disease) (Bourdes, et al. 2011).

Sensitivity measures the fraction of positive cases classified as positive. Specificity measures the fraction of negative cases classified as negative.

AUROC is a good overall measure of predictive accuracy of a model. It represents the area under the ROC curve, a measure of how well a model can distinguish between disease and normal groups. An AUROC value near 0.50 suggests no discrimination, namely one can flip a coin to decide. But, an AUROC close to 1.0 is considered excellent discrimination (Linder, Geier, Kolliker 2004). The single measure for accuracy comparison of models and committee of models will be AUC, Area Under Characteristic curve.

2.4.2 Well-posedness: Stability & Immunity to small perturbations of input data

Generally a mathematical model is regarded well-posed if it meets Hadamard's three criteria: 1) the model has a solution, 2) the solution in unique, and 3) the solution depends continuously on the data (Lucchetti 2006). Conversely an ill posed mathematical model has initial, or boundary data, where an infinitesimal perturbation can grow unbounded away from the unperturbed solution. Generally, if it can be proven that the solution is uniformly bounded everywhere then it is well posed. But on the other hand it's possible to have unbounded solutions which are not ill posed. Since most classification problems include local optima as possible solutions, they're not regarded as well-posed. However to overcome this limitation, I've applied genetic algorithm version of ANN such that the solutions consider multiple optima and avoid the local optima trap. Therefore some level of regularization is

necessary for the other models. In other words, one must know how to use additional assumptions to create a well-posed behavior in these Artificial Neural networks.

2.4.3 Utility: Practicality in the medical workflow

Once the model is trained, the model can be set up to run automatically at certain time intervals ranging from every minute to every several hours. It's practical to have all four models trained in advance and run them in parallel. An Oracle program can provide the most accurate prediction by polling the four ANN model results. The results can be filtered by the Oracle such that if the occurrence of a disease is detected, the Oracle program would send an alert to the physician.

2.4.4 Adaptability: Ability to handle new evidence, i.e. new data values and data types

ANNs are able to adapt to new data sets, additional variables and all data types. It's recommended that an ANN model be retrained after every few months to adapt to new data and overcome the temporal, environmental and demographic changes that might occur in patients. In this research, the Memory module collects and maintains clinical data. ANN models can be re-trained as new data become available from the Memory module.

2.4.5 Economy: Cost of computation and timeliness of prediction

In computing, the computational cost of algorithms is determined by an asymptotic number of computations required for an algorithm to complete. The complexity measure of neural network algorithms provides an upper limit on the worst case scenario when the input variables and number of cases grow large. The computational cost of an algorithm is a function of number of steps to compute (time complexity), memory size (space complexity) and length of algorithm. In ANNs, the number of computations is a time complexity, the number of perceptrons is a measure of space complexity and the number of weights is a measure of algorithm length. The complexity of ANNs has been shown to be NP-complete, namely given enough information and hardware, they can predict any input-output function in a finite time (Kon, Plaskota 2000).

The four ANN models used in this study were trained in under 5 minutes of CPU time and under one hour elapsed time. Although an ANN software package was used for this research, the algorithms are available for programming and one could develop their set of

four ANN models using a common programming language. The cost of computation is not extreme for a new prediction and can be completed within less than 2 minutes for cases of comparable dataset sizes, such as the case studied in this research.

The development of the four models from a prognostics perspective are explained in the next chapter.

Chapter 3

3 PROGNOSTICS MODEL AND FRAMEWORK DEFINTION

3.1 Prognostics & Health Management as a Discipline

Prognostics is the science of predicting the future functionality of a system by estimating the remaining useful life, probability of failure or time to failure for a given system. The root of the word "system" comes from the Latin word "systema" and the Greek word "systema", meaning a whole compounded of several parts (Merriam-Webster 2011). The closest engineering definition for a system is given as "a group of interacting, interrelated, or interdependent elements forming a complex whole" (Dictionary.com 2012). According to Zadeh and Desoer, to study a system, one defines a simpler representation in form of a model (or models) that represent the physical system (Zadeh and Desoer 1963). Given a model, a mathematical representation and notation for the system can be developed. Finally one can analyze the model by considering its properties, capabilities and its limitations; these three goals are regarded as the task of system theory (Zadeh, Desoer 1963).

Control theory involves the study of "control" or "regulation" of an object or process. Control systems can be classified into either "open" or "closed" systems. In the "open-loop" system the goal is to program the system in advance to give a desired output. This is the notion of a feed-forward mechanism. The closed-loop system relies on feedback from the system's output in order to make adjustment to the input so the system gives the desired output (Wishart 1969).

A control system consists of certain variables that affect the system, the output of the system called output, a controller that compares the input with the output and uses the difference to activate the control elements. The signal that returns the output to the controller is called feedback.

In an open loop system, the goal of properly regulating the system can be attempted by using a feed-forward signal. Since it's possible to monitor and measurement certain variables of the system, one can use those measurements to make adjustments the input (instead of adjusting the control elements) to ensure the system gives the desired output.

In control theory, an interesting question is the level of stability that can be achieved, and optimal control. In this research a hybrid control system is proposed, consisting of both

feed-forward and feedback control. Our goal is to apply control theoretic approach to prognostics. Prognostics is the science of predicting the future functionality of a system by estimating the remaining useful life, probability of failure or time to failure for a given system or component in the system.

3.2 Control Model

In this paper, notations consistent with control system research are used (Kapur 2010). A typical system consists of input ***r*** and response variable (or output) represented by ***y***. If the researcher has perfect knowledge about this system, and knows the transfer function $f_o(r) = y$, then inputs ***r*** can be determined as $r = f_o^{-1}(y)$. Assuming the system (transfer function) is known, one can predict the response variable ***y*** and adjust the inputs ***r*** to maintain the output within the desired range. This is the ideal process as shown in Figure 4.

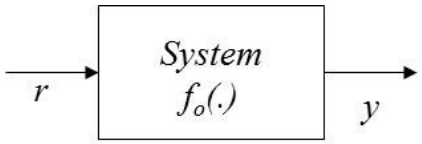


Figure 4: The ideal "Perfect" Process

A perfect prognostics is the situation where the researcher knows the transfer function and a perfect knowledge of the system is available. Since the desired output ***y*** is known, one can determine how input ***r*** should be adjusted. However, there are challenges to achieving this goal:

1. The inverse problem is not unique and not easy to determine.
2. There is often lack knowledge (or there is uncertainty) about our model.
3. The real world systems might be very complex and cause output ***y*** to appear as random variable.

In real world, not all systems offer a perfectly known transfer function. Since the causes of variation can't be perfectly known, one must attribute the variation in ***y*** to disturbance represented by ***d***, as shown in Figure 5.

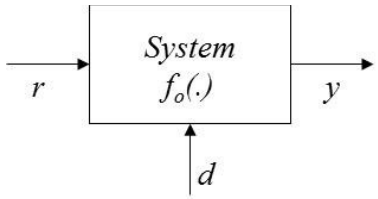


Figure 5: Process with disturbance *d* causing variability in output *y*

One way to correct for the effects of disturbance and variation is to use feedback as shown in Figure 6. Feedback signal is represented by signal ***v***. In a basic feedback loop, the output is returned back to a controller to adjust the input. But using feedback to adjust input proves to be too late for adjusting the input in a timely manner.

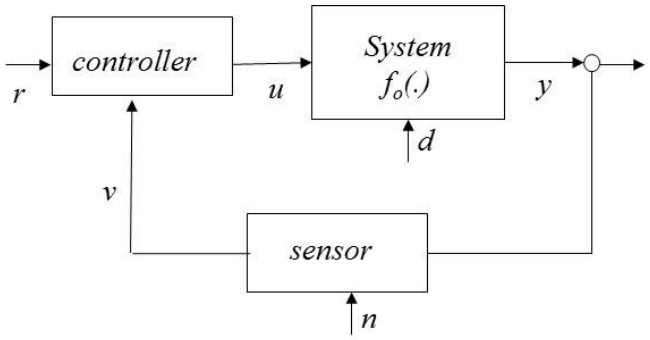


Figure 6: Process with feedback loop to reduce variability in output *y*

The traditional approaches have been based on feedback to correct the system behavior. However, instead of being reactive, it's more desirable to be proactive and prevent deviations in the first place from ideal or target value to occur in the response ***y***. Prognostics models use feed-forward and develop a functional relationship between input variables and adjust the input variables to achieve the desired results.

In prognostics the intent is to understand the underlying causes of error (variation or uncertainty based on empirical, incompleteness, ambiguity, fuzziness, vagueness, etc.) as much as possible. One can decompose disturbance ***d*** further to sub-components, namely to disturbances due to other factors (call it ***Z***). Thus disturbance ***d*** can be represented by two variables: a part that can be measured and understood, let's call it ***Z***, and by ***e*** which is the remaining error and unknown cause of variation on output ***y***. This is illustrated in Figure 7. Thus, now the error term can be written as:

$$\boldsymbol{d} = \{\boldsymbol{e}\ \ , \boldsymbol{Z}\} \tag{9}$$

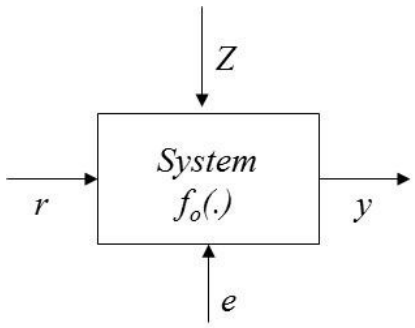


Figure 7. System with decomposition of error *d* into *e* and *Z*

When the system is shown with prognostics P and prediction Z, one can introduce an intervention process to modify the input u into the system as shown in Figure 8.

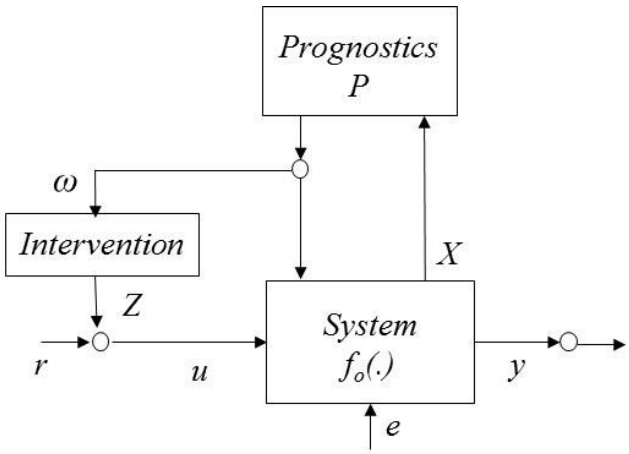


Figure 8: Prognostics *P* with feed-forward *ω*

Therefore, the entire disturbance can be explained and measured partly by ***Z*** and by the remaining disturbance or error represented by ***e***. One can measure this disturbance ***Z*** (though it's not possible to change or control disturbance) to determine how to change the input variables to create feed-forward and maintain the system response variable ***y*** closer to the target. In this model, ***X*** is the measured data collected from the system by monitoring certain internal variables.

The goal is to expand on this notion by including the role of three additional components; Monitoring, Memory and Intervention modules in this feed-forward model. In Figure 9, a process for monitoring and collecting measurement data is shown that's represented by ***X,*** and obtained from the system. The goal of the Prognostics engine is to determine ***Z*** based on data collected by the monitoring module and prior historical data collected in a memory module. The prognostics engine applies logic to both the measured

and historical data represented by ***h***, to determine ***Z.*** The memory module is passive as it stores historical data and provides that data upon recall by the Prognostics engine.

The monitoring module is subject to noise represented by ***n***. The input value to the System is represented by ***u*** as shown in Figure 9.

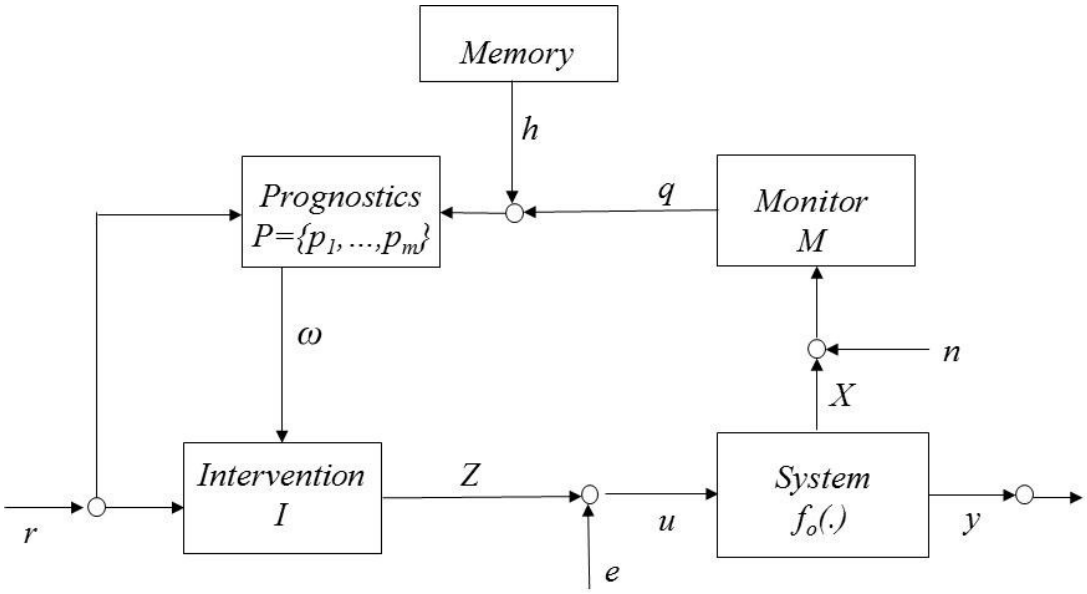


Figure 9: Feed-forward process with Monitor, Prognostics, Memory and Intervention

The input-output signals have the following interpretations:

r	reference or command input
u	System input
e	External disturbance
y	System output
X	Set of monitored and measured signal collected
n	Sensor noise
q	Measured signal, input to Prognostics engine
h	Historical measurements
Z	Prognostics output
P	Set of Prognostics rules

The three signals coming from outside- ***r***, ***n*** and ***e*** –are known as exogenous inputs.

In the model above, we're interested in well-posedness, namely all transfer functions exist and produce the outputs from the three exogenous inputs.

Next, consider a feed-forward-feedback control model, by taking the output signal through a controller (or observer) and feed it back into the input. This model is shown in Figure 10. An observer process converts the output into signal ***v*** and feeds it back to an operator that either adds or subtracts from the input signal.

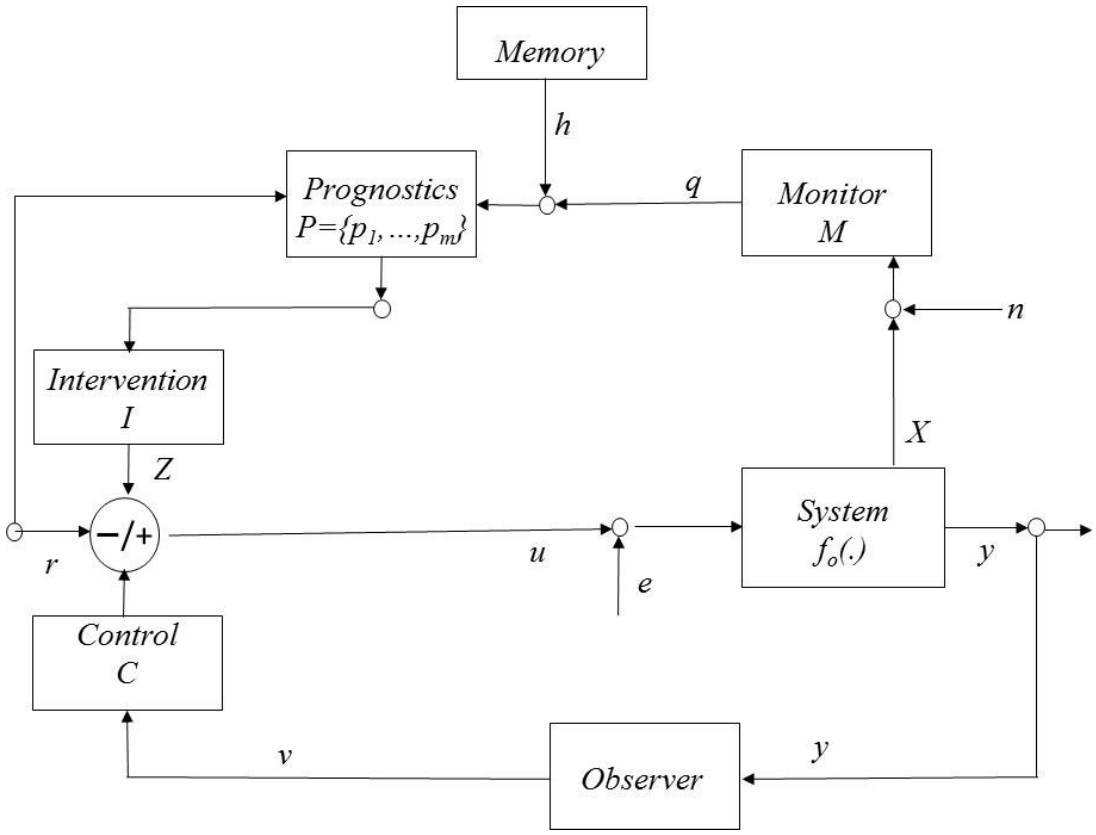


Figure 10. Combined Feed-forward and feedback model with input, output signals

3.3 Why Feed-forward Control?

There are two potential issues associated with feedback control: First, the time lag to receive the output back into the controller is often too long and as a result there is insufficient time to correct the input. Second, there are situations where the rate of change in disturbance and deviation from a target range of output is slight and gradual over a long time. Finding the exact correction value to the input in order to cancel the disturbance in such systems can be difficult. In several systems, including human health system these conditions can be present where either the onset of an ailment is sudden which does not afford adequate time to respond, or the patient's health decay are too gradual to notice until a sudden change occurs.

Feed-forward control on the other hand offers the ability to apply corrections before the system output falls out of the desired range. Combining feed-forward plus feedback control provides significant improvement over feedback control.

The result of timely prediction and intervention enables physicians to reduce the occurrence of medical complications such as DVT through prevention. For illustration refer to the conceptual graphs in Figure 11. Graph (a) represents a conceptual frequency of DVT cases that occur without predictive tools. In contrast, graph (b) represents a smaller and delayed frequency of cases as a result of earlier prediction and intervention.

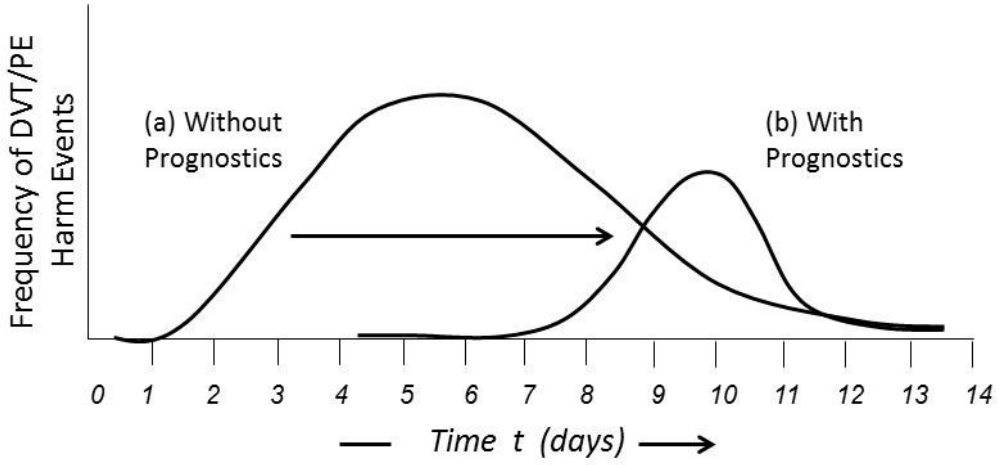


Figure 11: Harm event distribution graphs with and without prognostics

3.4 Why ANNs

Of the various statistical and computational methods covered in predictive medical models, Artificial Neural Networks offer unique advantages that make them a suitable tool for research and prognostics. These advantages outweigh some of the criticisms that have been leveled against ANNs (TU 1996).

The primary criticisms of ANNs include a "black-box" approach to data, proneness to over-fitting and greater computational burden. Requiring greater computational power is less of an issue now as the desk top and portable computers have much more powerful computational power. The criticism about "black-box" approach is not a serious limitation in this research since multiple models are used and supervised learning is applied. Over-fitting is a weakness that occurs when a model is trained to a specific data set and performs poorly on other datasets not used in training. This weakness can be avoided through multiple

iterations of cross-validation and setting aside a separate test data batch as employed in this research and explained in Chapter 4.

In contrast the advantages and reasons for choosing ANNs as predictive models are significant considerations:

- Ability to model complex non-linear relationships between input data and output
- Ability to learn and adapt to patterns in data
- Resilience towards missing data elements
- Many algorithms are available to choose from
- Ability to handle a large amount of variables
- Ability to handle diverse types of data
- Ability to detect all possible interactions among predictor variables

3.5 Introduction to ANNs

This study applied and compared prediction results from all four neural network models. Neural networks have been successfully applied to classify patterns based on learning from prior examples. Different neural network models use different learning rules, but in general they determine pattern statistics from a set of training examples and then classify new data according to the trained rules. Stated differently, a trained neural network model classifies (or maps) a set of input data to a specific disease from a set of diseases.

Artificial Neural Networks (ANNs) are inspired by the biological learning processes. ANN models are parallel information processing constructs that attempt to mimic certain biological neural systems. ANN offer many advantages: they can model both linear and non-linear problems. They can scale up and down depending on the size. Their parallel construct provides self-healing and redundancy. Models based on ANN constructs attempt to answer several questions about learning, classification and pattern recognition. These attributes are useful features of cognitive and reasoning that occur in medical decision making.

The goal of ANN is to mimic the nervous system. Just as the nervous system consists of an interconnection of simple units, called nerve cells, an ANN consists of many independent but inter-related elements (called neurons) organized into layers. Each neuron transmits an excitation or inhibitory signal to another neuron. The contribution of the signals depends on the strength of the synaptic connection. Similarly, biological neural learning

happens by the modification of the synaptic strength. In a neural network the synaptic strengths are represented by weights associated with each input.

A typical ANN is composed of layers connected to each other by full or random connections. There are typically two layers with connection to the external world: an input layer where data is collected and an output layer that presents the outcome or response of the network. But multi-layer ANN models are common. Figure 12 shows the a simple neuron consisting of input signals designated by $x_1, \ldots, x_k$, weights associated with each signal $w_{i1}, \ldots, w_{ik}$, a summing junction and an activation function that produces output Y_i.

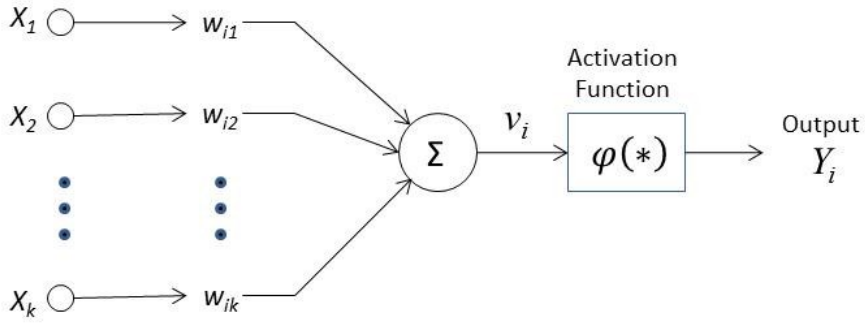


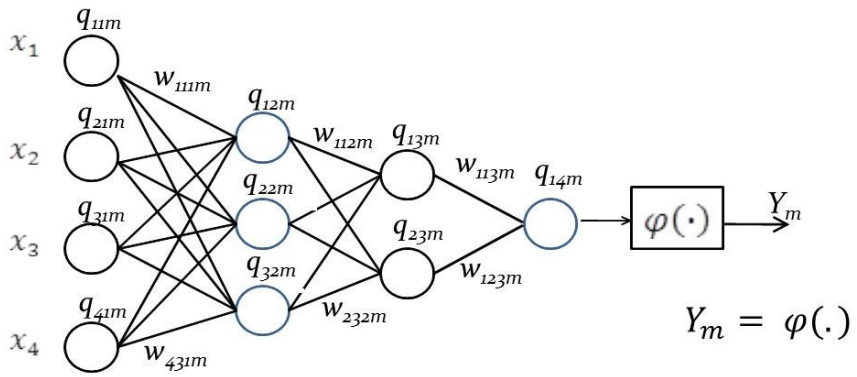
Figure 12: A simple Neuron with activation function for node *i*

For each neuron, the summation function aggregates a weighted sum of inputs while the activation function transforms the sum into the final output of the neuron. The formula of each step is shown below:

$$v_i = \sum_{j=1}^{k} x_j w_{ij} \tag{10}$$

$$Y_i = \varphi(*) \tag{11}$$

In Figure 13 the general structure of a multi-layer ANN is shown. The data gathered about a patient's condition is fed into the model through a layer of neurons. Here four input signals are shown. The result of each layer is an activation function whose output is input to the next layer. There are m rules in the framework, each detecting a particular disease. Layers are shown by neurons q_{jpm} and weights that connect layer $(p+1)$ to layer p by w_{ijpm} , where i denotes the number of neuron in layer p and j the number of neuron in layer $(p+1)$; the subscript m denotes the perceptron parameters for model m.



where q_{ipm} *, for p=1 to n, n=4 is the neuron no. i at layer p for model m, and* w_{ijpm} *, for p=1 to n-1, n=4 is the weight of neuron j to neuron i in layer p for model m*

Figure 13: A 4-layer Neural Network

Then the equation to calculate the value of every neuron in each layer in the *n*-layer network above can be described as:

$$q_{i(p+1)m} = \sum_{j=1}^{k-p+1} q_{jpm}\, w_{ijpm} \text{ , for } p=1, \dots, n-1, \text{ and layers, } i=1, \dots, k-p \qquad (12)$$

where *k* is the number of input measurements *x(t).* Some well-known documented advantages of ANN are learning and pattern recognition. Depending on the activation function, the final output can be a "1" or "0" indicating whether the patient is in danger of developing DVT/PE symptoms or not. Among the training methods, Back-propagation is a common technique for training neural networks. This research used this technique to train a model and applied it to new set of patient data for predictive purposes.

Back-propagation consists of two steps: In step one; the researcher calculates error contributions to the response function *Y*. This step computes how much each neuron has contributed to the total error in the response value. Error is defined as the difference between the ideal (or expected) result versus the actual response value. Neurons with higher weights have contributed more to the total error and therefore their weight needs to be adjusted more. In step two, the algorithm adjusts the weights starting from the outer layer neurons going back to the hidden layers finally reaching the weights of the input layer. When this algorithm completes, the network has been trained. This is called supervised learning because it defines the ideal (or expected) response value.

Once the neural network model is trained, it can be applied to a fresh or incomplete set of data. The outputs will provide predictions based on the inputs and adjusted weights.

3.6 A Simple Example

The following are two examples of simple, single-layer perceptron classification. These examples are adapted to this research. The original examples appear in Zurada (Zurada 1997), Haykin (Haykin 1998), Sengupta (Sengupta 2009), and Masters (Masters 1995). One can classify input data about patients into two categories of predictions: DVT-True and DVT-False, by looking at prior patient data. The objective of the single-layer perceptron is to determine a linear boundary that classifies the patients on either side of the linear boundary. As shown in Figure 14, the intent is to classify patients into two categories separating by a boundary called a decision boundary line. A linear set of equations define this boundary. The region where the linear equation is >0 is one class (DVT-True), and the region where the linear equation is <0 is the other class (DVT-False). The line is defined as:

$$w_1 x_1 + w_2 x_2 + w_0 = 0$$

One can apply a threshold function to classify patients based on the following threshold function:

$$p(x_1, x_2) = \begin{cases} 1 & if\ w_1 x_1 + w_2 x_2 + w_0 \geq 0 \\ -1 & if\ w_1 x_1 + w_2 x_2 + w_0 < 0 \end{cases}$$

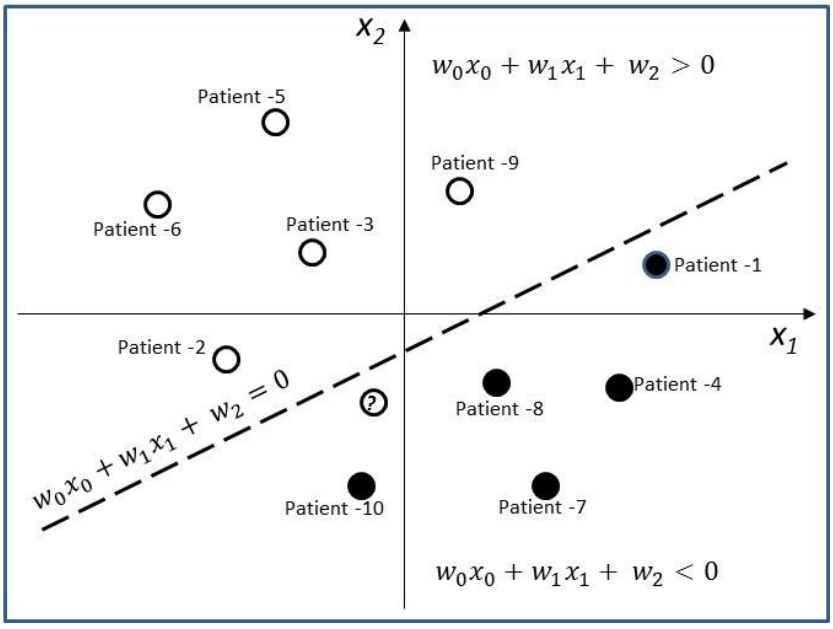


Figure 14: Classification using single-layer perceptron

Suppose we're considering classifying patients by only four input variables, Glucose (G), Body mass (M), Systolic Blood pressure (S) and White blood cell count (B), represented by x_1, x_2, x_3, and x_4. The threshold function would be computed as follows:

$$p(x_1, x_2, x_3, x_4) = \begin{cases} 1 & if\ w_0 + \sum_{i=1}^{4} w_i x_i \geq 0 \\ -1 & if\ w_0 + \sum_{i=1}^{4} w_i x_i \leq 0 \end{cases}$$

Let's assume the following weights and input values for classification example are given as shown in Table 1:

Table 1: Computing classification using single layer perceptron classification

Weights	Values	Inputs	Values
w_1	2	x_1	-1
w_2	0	x_2	2
w_3	3	x_3	0
w_4	-1	x_4	-4
w_0	1	*Bias*	1
$p(x_1, x_2, x_3, x_4) =$ 2*-1 + 0*2 + 3*0 + -1*-4 + 1*1 = 3 ⇒ class = 1 or DVT-True			

DVT is the name of the disease. If this classification is incorrect, then it's necessary to adjust the weights and repeat the process until the patient is correctly classified. Suppose the correct classification is (-1), then the calculation proceeds as shown in Table 2. The results indicate which class the data belongs to, which in this example the classification is no-disease or DVT-False.

Table 2: Revising weights to correct misclassification

Weights	Values	Inputs	Values	New Weight calculation when actual *class = -1*
w_1	2	x_1	-1	$w_1 = w_1 + class * x_1 = 2 + (-1) * (-1) = 3$
w_2	0	x_2	2	$w_2 = w_2 + class * x_2 = 0 + (-1) * 2 = -2$
w_3	3	x_3	0	$w_3 = w_3 + class * x_3 = 3 + (-1) * 0 = 3$
w_4	-1	x_4	-4	$w_4 = w_4 + class * x_4 = -1 + (-1) * (-4) = 3$
w_0	1	*Bias*	1	$w_0 = w_0 + class * x_0 = 1 + (-1) * 1 = 0$
$p(x_1, x_2, x_3, x_4) =$ 3*-1 + -2*2 + 3*0 + -1*3 + 0*1 = -7 ⇒ class = -1 or DVT-False				

3.1 A Simplified Mathematical Example

In this section, a simple ANN model is presented as an example using the XOR logic table for illustration. As shown in Table 3, the XOR table returns value of 0 if both inputs are identical (both 0's or 1's) and returns value of 1 if one or the other input is a 1.

Table 3: The XOR Logic Table and Results from ANN Model

Input x_1	Input x_2	Ideal Value
0	0	0
0	1	1
1	0	1
1	1	0

It's possible to develop a 3-layer neural network to compute the result for each pair of inputs x_1 and x_2. A third input called Bias is also introduced to construct the model. The value of Bias is always 1. In the first iteration random weights are used. There are a total of 9 weights in this model as shown in Figure 15. Simply put, the output is a function of weights and inputs. This is an example of supervised learning as the weights in the ANN model get trained to produce the desired output.

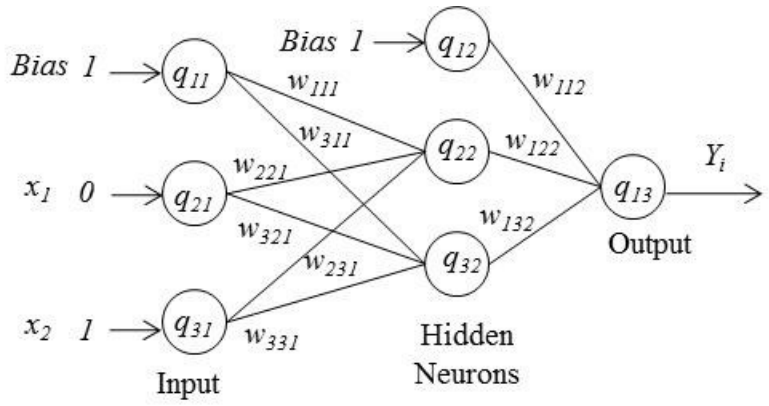


Figure 15: A 3-layer ANN Network to Compute the XOR Logic Table

The goal is to adjust the weights iteratively until the ANN model produces the Ideal Value. In the first iteration, the model produces some results shown in the Output column. The model computes the error as Mean Square Error (MSE) and uses the error to adjust the weights. The iterations continue and weights get adjusted until the error term is below a threshold (in this case less than 0.009). Eventually the ANN model stops and the output is the Final result as shown in the last column in Table 4. The computed results are close to the ideal values (close to 0 or 1), only different by a small margin of error.

Table 4: The XOR Logic Table and Results from ANN Model

Input x_1	Input x_2	Ideal Value	Output	$(\text{Error})^2$	Final Results
0	0	0	0.2	0.04	.00875
0	1	1	0.3	0.49	.99130
1	0	1	0.4	0.36	.99123
1	1	0	0.5	0.25	.00568

3.1.1 Activation Functions

One of the key features of Artificial Neural Networks is that one can map a linear neuron output into a non-linear activation function (Haykin 1998, Sengupta 2009, Zurada 1997). Given inputs $x_0, x_1, \ldots, x_n$, the output v_k is the result of summation from Equation (10) and $Y_k = \varphi(*)$ is the result of the activation function. The activation function results in an S-shaped curve known as the sigmoid function. There are three types of activation functions. In the first type, as v_k changes from $-\infty$ to $+\infty$, the output can vary from 0 to 1,

namely $y_k = [0,1]$. This is the logistics function shown in Figure 16. The activation function for this type of neural network is shown as:

$$\varphi(*) = \frac{1}{1 + \exp(-av)} \tag{13}$$

$$\varphi(*) = \begin{cases} 1 & when\ v \to +\infty \\ 0 & when\ v \to -\infty \end{cases} \tag{14}$$

This is the simplest neuron formulation. It's possible to change the shape of the S-curve by changing the values of a. When a is small, the curve appears as smooth function. But, when a is very large, this function approaches the threshold function, a model proposed by Pitts and McCollough (It's also known as the Pitts-McCollough model).

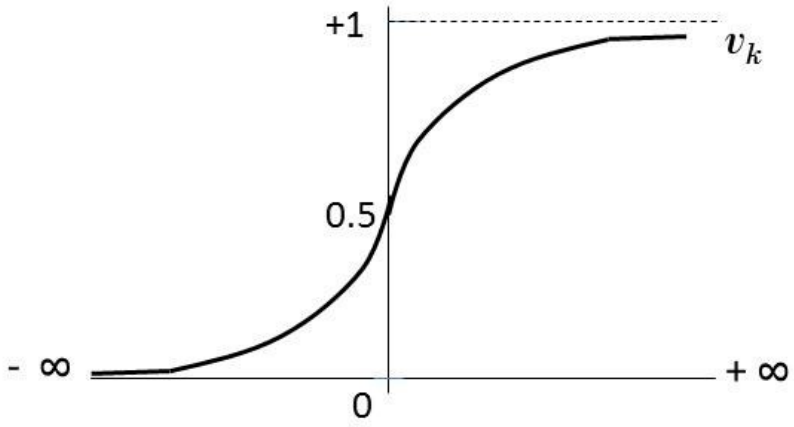


Figure 16: Sigmoid function for activation function v_k between [0,1]

The second type of activation function has a range from -1 to +1, shown with the following equation:

$$\varphi(v) = \tanh(av) \tag{15}$$

The tanh(.) is the hyperbolic tangent function computed as follows:

$$\tanh(v) = \frac{e^{v} - e^{-v}}{e^{v} + e^{-v}} \tag{16}$$

The activation function $\varphi(v)$ is determined according to the value of v:

$$\varphi(*) = \begin{cases} 1 & when\ v \to +\infty \\ -1 & when\ v \to -\infty \end{cases} \tag{17}$$

The shape of S-curve representing this activation function is shown in Figure 17.

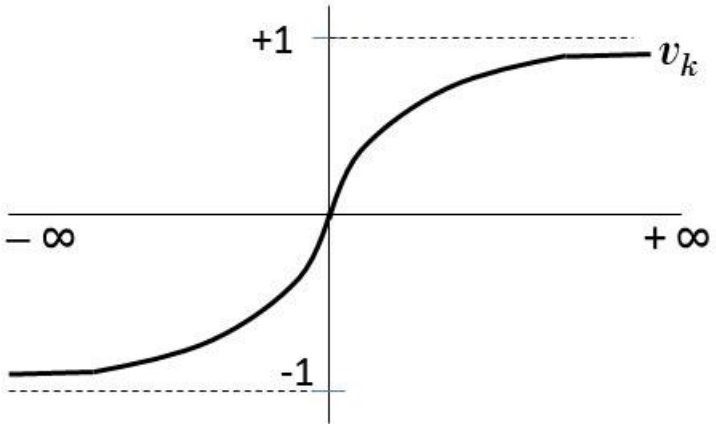


Figure 17: Activation function for v_k between [-1, 1]

The third type of activation function is the stochastic model determined by:

$$\varphi(v) = \begin{cases} 1 & with\ prob\ p(v) \\ 0 & with\ prob\ 1 - p(v) \end{cases} \tag{18}$$

and $p(v) = \frac{1}{1 + e^{(-v/T)}}$ (19)

When $p(v) = 1$ then $T = 0$ and this activation function becomes a deterministic model. As *T* gets larger, there is more stochastic behavior in the model. To better illustrate the role of *T*, it's possible to think of it as temperature or kinetic energy borrowing this concept loosely from the third law of thermodynamics. Figure 18 shows the S-curve associated with this activation function. The various S-curves illustrate the effect of *T* on the values on the curve.

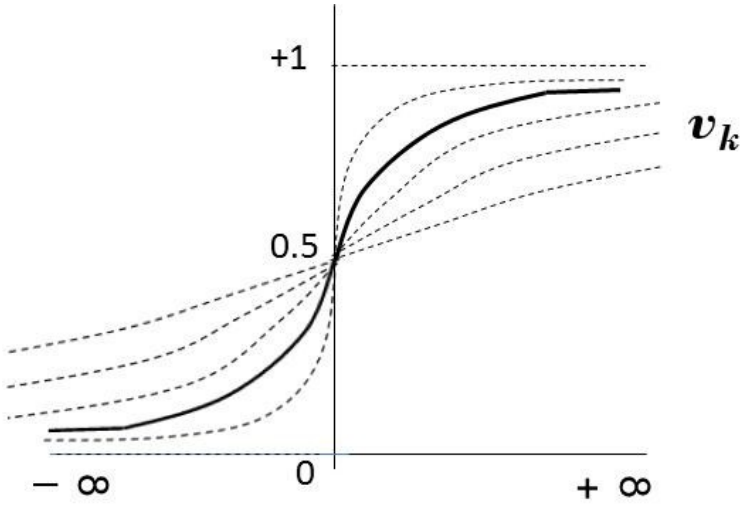


Figure 18. Stochastic Activation Function for v_k

3.2 Mathematical Foundations of Artificial Neural Networks

A mathematical foundation of ANNs is presented in this section using the common conventions of ANN formulations, so inputs are represented by x, the desired output by d, weights by w, and ANN's output by y (Zurada 1997, Haykin 1998, Wang 2012, Sengupta 2009). As introduction a simple neuron is presented followed by formulations for classification, memory and learning. A single neuron can be constructed with a single activation function. Consider finding a regression line for the histogram shown in Figure 19. The regression line is represented by

$$y = mx + b \tag{20}$$

Where m is the slope of the line and b is a constant, known as bias. The corresponding neural network representation uses a single neuron where weight parameter w_{11} corresponds to slope m and w_{10} is equal to a constant 1.0.

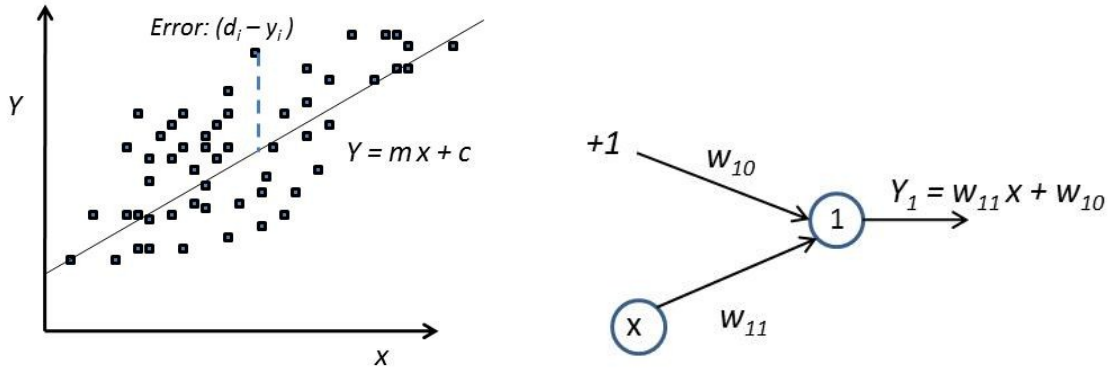


Figure 19: A regression line and its equivalent single neuron representation

If y is dependent on multiple inputs x_j, one can think of bias as another input to the neuron with a weight w_{k0}. To find the best fitting line, the goal is to minimize the errors E, by adjusting weights. Given multiple x_j one can use the gradient descent method to find the minimum E and the corresponding weights.

3.2.1 Gradient Descent

The Gradient Descent method is used as an iterative process to determine the weights associated with each input x. This method which is also known as Error Correction learning in ANN literature works to minimize the total error as the method of training the model and determining the appropriate weights. The following derivation is adapted from Zurada

(Zurada 1997) and Sengupta (Sengupta 2009), improved and revised specific to this research. Total error E can be written as:

$$Total\ Error\ E = \sum_j E_j = \frac{1}{2}\sum_j (d_j - y_j)^2 \quad (21)$$

This represents the total error E for point j. The expression d_j is the target output (desired output) at point j, and y_j is the actual output at point j.

Let's now consider all possible outputs $y_0, \ldots, y_m$ where $0 \le j \le m$, and

$$y_0 = f_0(x_1, x_2, \ldots, x_n) \quad (22)$$
$$y_1 = f_1(x_1, x_2, \ldots, x_n)$$
$$\ldots$$
$$y_m = f_m(x_1, x_2, \ldots, x_n)$$

Total error E is the combined error of all errors for outputs y_k. It's common to use ½ of the sum in (14) since as will be explained later it makes mathematical manipulations easier as one takes derivative of this term and the gradient will be multiplied by 2.

Let's define the gradient, namely the rate of increase for (i,j) pair connection as:

$$G = \frac{\partial E}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}} \sum E_j = \sum_j \frac{\partial E_j}{\partial w_{ij}} \quad (23)$$

Next, it's possible to apply partial derivatives and chain rule to get the following:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_j} \cdot \frac{\partial y_j}{\partial w_{ij}} \quad (24)$$

For sake of simplicity let's denote d_j and y_j as follows:

$$d_j = \sum d_j \text{ , and } y_j = \sum y_j$$

One can take derivative of the total Error, Equation (21) to get the following:

$$\frac{\partial E}{\partial y_j} = -(d_j - y_j) \quad (25)$$

$$y_j = \sum_j w_{ij} x_j \quad (26)$$

$$\frac{\partial y_0}{\partial w_{0i}} = \frac{\partial}{\partial w_{0i}} \sum_j w_{oj} x_j = x_i \quad (27)$$

$$\frac{\partial E}{\partial w_{ij}} = -(d_j - y_j) x_i \quad (28)$$

Where j is the output unit, and i is the input unit. Thus the derivative of error with respect to w_{ij} has been formulated. In order to move the opposite direction to the derivative one can

apply the corrections to the w_{ij}'s by multiplying a (-) sign to the difference. The (-) sign is applied because the goal is to minimize error. The correction can be written as:

$$\Delta w_{ij} = (d_j - y_j)x_i \quad (29)$$

The new synaptic weight will be computed using the following for several iterations until Δw_{ij} is less than a given threshold set by the user:

$$w_{ij_{(new)}} = w_{ij_{(old)}} + \Delta w_{ij} \quad (30)$$

It's possible to use η to represent the rate of descent in (29). So η is the learning rate that reduces total error *E*, with every iteration and can be defined by the researcher to regulate Δw_{ij}, the rate of descent.

3.3 Neural Network Learning Processes

There are five major categories of learning models in Neural Networks. One of these learning methods called Error Correction based learning was already discussed in section 3.2.1. In this section, four other categories are presented that are most relevant to this research: Memory based learning, Hebbian based learning, Competitive learning and Boltzman learning model. These learning methods are adapted from Sengupta (Sengupta 2009), Zeruda (Zeruda 1997), Wang (Wang 2012), Masters (Masters 1995) and Haykin (Haykin 1998). They're refined and revised for this research and are included for the sake of completeness.

3.3.1 Memory Based Learning

Memory based learning works to retain relationship between input vector and output. Given input vector $\vec{x}$defined by $\{x\}_{i=1}^{N}$, and desired output d_i, this association can be shown by the expression $\{x_i, d_i\}_{i=1}^{N}$. When the model is applied to a new pattern $\vec{x}_i$, since this pattern is initially unknown let's start with a test pattern $\vec{x}_{test}$ and find the Euclidean distance between $\vec{x}_{test}$ vector and the new pattern $\vec{x}_i$ vector. Let's assume that $\vec{x}'_N \in \{x_1, x_2, x_3, \dots, x_N\}$ is the set of nearest neighbor points of $\vec{x}_{test}$ vector, then it implies that the distance of pattern $\vec{x}_i$ from $\vec{x}_{test}$ is minimum over the set of all $\vec{x}_i$. This can be shown by the following expression to be true for all distances over *i*:

$$\min_{(over\ i)} \quad d\{x_i, \vec{x}_{test}\} = d(\vec{x}'_N, \vec{x}_{test}) \quad (31)$$

To improve this algorithm it's prudent to look at the nearest neighbors and find the set that offers minimum distances. Memory based learning is ideal for pattern recognition and classification of data as shown by example in Figure 20. This approach helps keep outliers out of classification. This is the k-nearest neighbor classification. In Figure 20, a point x_i is being classified between the "+" or the "O" shapes. Since its nearest neighbors are the "O" shapes, it will get classified as a member of the "O" set.

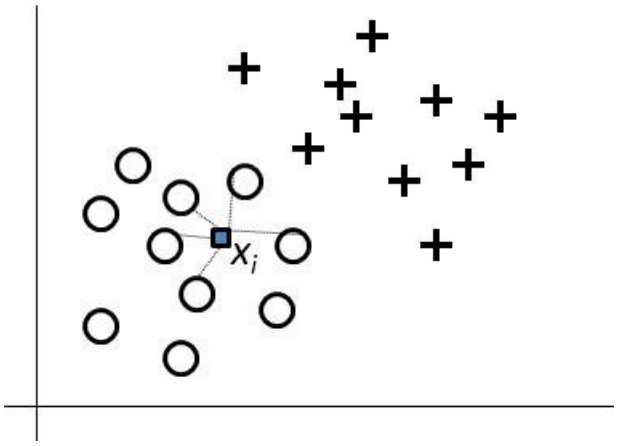


Figure 20: Classification using Memory Based Learning

3.3.2 Hebbian Based Learning

The goal of the Hebbian based learning is to retain the association between the input vector and the output. This method is attributed to Donald Hebb, a neurobiologist who in 1949 introduced his theories of neuron adaptation in the brain during the learning process. In the Hebbian learning process the amount of adjustment to weight w_{kj} is defined by:

$$\Delta w_{kj}(n) = f(y_k(n), x_j(n)) \tag{32}$$

This is the adjustment in w_{kj} at time step n, as a function of responses y_k and input x_j at time step n. One can re-write this expression in terms of pre-and post-synaptic responses:

$$\Delta w_{kj} = \eta\, y_k(n)\, x_j(n), \tag{33}$$

where η is the rate of learning. This expression is known as the Activity Product Rule. It's important to note that η is constant. Assuming that one keeps x_j constant then it's possible to plot Δw_{kj} and $y_k(n)$ to get a line that intercepts through the origin with slope of $\eta\, x_j(n)$ as shown Figure 21. As y_k increases, so does Δw_{kj}. Eventually the synaptic weight reaches its saturation point where not more learning possible.

Let's define $\bar{x}$ and $\bar{y}$ as time averaged values of x_j and y_k. Then it's possible to define:

$x_j(n) = x_j - \bar{x}$, and $y_k(n) = y_k - \bar{y}$.

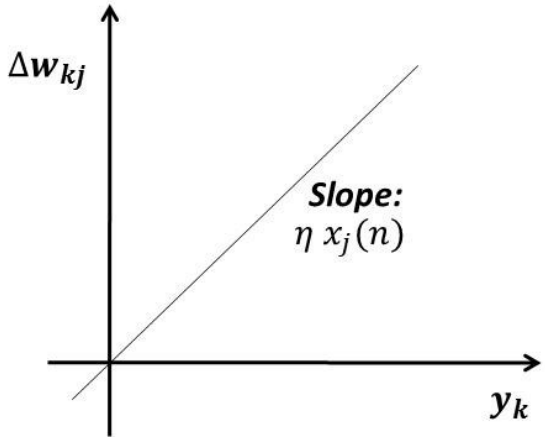


Figure 21: Slope of Activity Product Rule as rate of learning

By definition of covariance, it's possible to write the change in weights w_{kj} as the covariance of distance of x_j and y_k from their respective time averaged values $\bar{x}$ and $\bar{y}$.

$$\Delta w_{kj} = \eta\,(x_j - \bar{x})(y_k - \bar{y}) \tag{34}$$

Since the average effect of change over the entire input values of x_j is desired, one can recognize $\bar{x}$ as a constant over the course of x_j. This relationship can be shown by a line that intersects Δw_{kj} and y_k as shown in Figure 22. This figure shows the relationship between Δw_{kj} and y_k for a given point x_j such that $(x_j - \bar{x})$ is a constant.

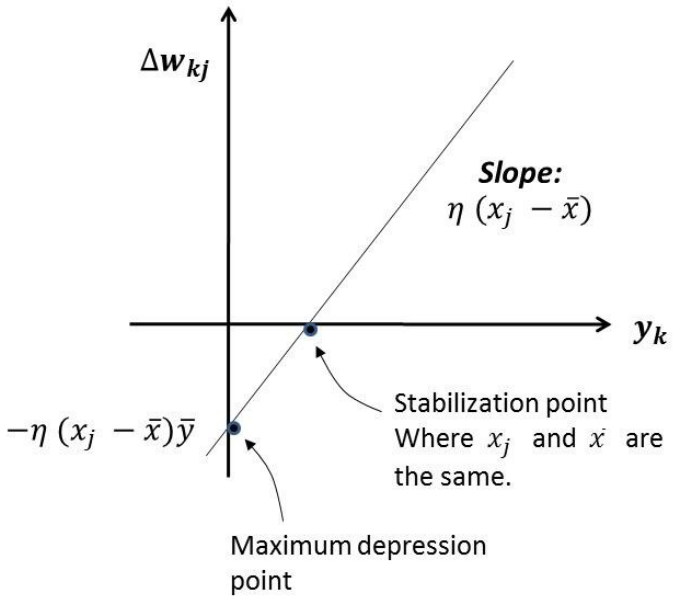


Figure 22: The covariance relationship between response and input

Using covariance approach, three conditions are possible:

$$(i)\quad w_{kj}\ increases\ if\ x_j > \bar{x}\ and\ y_k > \bar{y}$$

$$(ii)\quad w_{kj}\ decreases\ if\ either \begin{cases} a)\ x_j < \bar{x}\ and\ y_k > \bar{y} \\ b)\ x_j > \bar{x}\ and\ y_k < \bar{y} \end{cases} \tag{35}$$

$$(iii)\quad w_{kj}\ increases\ if\ x_j < \bar{x}\ and\ y_k < \bar{y}$$

3.3.3 Competitive Learning

In Competitive learning, each neuron competes to increase its response value while minimizing the other neuron's output. The wining neuron will be preferred in future iterations of learning. The mathematical model of competitive learning is based on:

$$y_k \begin{cases} 1\ \ if\ v_k > v_j\ ,\ for\ all\ j\ when\ j \neq k. \\ 0\ \ otherwise \end{cases} \tag{36}$$

The sum total of all weights are set to 1 for all k:

$\sum_j w_{kj} = 1,$

For example, consider three clusters of input variables as shown in Figure 23. One can write x_j as a vector $\vec{x} = [x_1, x_2, x_3]$. If the relationship $\|\vec{x}\| = 1$ is enforced, namely that if it's required that $\sqrt{{x_1}^2 + {x_2}^2 + {x_3}^2} = 1$, then there can be several vectors $\vec{x}_1,\ \vec{x}_2\ ,\ \vec{x}_3$ to represent different patterns as shown in Figure 23. The goal is to classify data into any one of *n* patterns. The clusters of patterns are grouped into set of data in vectors (in this example in vectors $\vec{x}_1,\ \vec{x}_2\ ,\ \vec{x}_3$) such that: $\sum_j w_{kj} = 1$, for all *k*. In addition, it's possible to show weights for each cluster as a vector. For example, the vector of weights for the first cluster can be shown as: $\vec{w}_1 = [w_{11}\ \ w_{12}\ \ w_{13}\ ]$.

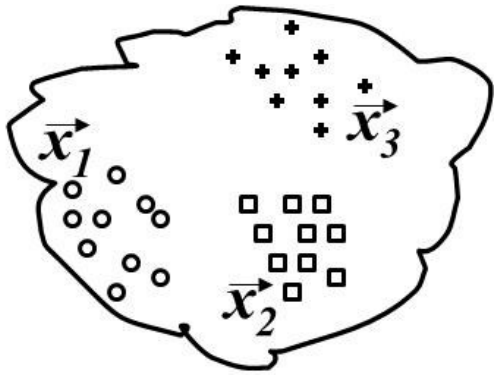


Figure 23: Using competitive learning to classify data into different patterns

In this competitive learning model, typically the most central element (or neuron) in a cluster is the winner and all other weights conform (or align) to it. The competitive learning rule is that Δw_{kj} is determined by:

$$\Delta w_{kj} = \begin{cases} \eta\left(x_j - w_{kj}\right) & if\ neuron\ k\ wins\ the\ competition \\ 0 & if\ neuron\ k\ loses \end{cases}$$

3.3.4 Boltzman Learning Model

The Boltzman learning process is derived from statistical mechanics and mimics a stochastic learning model. In this model the neurons constitute a recurrent structure that allows self-feedback. The neurons take a binary value of +1 or -1. The model is represented by:

$E = -\frac{1}{2}\sum_j \sum_k w_{kj} x_k x_j$ where $j \neq k$

The visible neurons are output layer. The inner neurons as shown in Figure 24 are hidden neurons. This model is regarded stochastic as a change in one outer neuron changes the value of E. The probability that a neuron x_k flips it state from one state to another state is defined by:

$$P\left(x_k \rightarrow -x_k\right) = \frac{1}{1+\exp(\frac{-\Delta E_k}{T})}$$

The change in E_k from a flip is denoted by ΔE_k. If E is be regarded as an energy function, then ΔE_k is the change of energy from a flip of state in a neuron. The variable is the pseudo temperature representing the level of noise or stochasticity. Let's assign two variables:

P_{kj}^{+}: the correlation between neuron k and neuron j in the clamped condition

P_{kj}^{-}: the correlation between neuron k and neuron j in the free condition

Then the Boltzman learning rule is defined by:

$\Delta w_{kj} = \eta\left(P_{kj}^{+} - P_{kj}^{-}\right)$ where $j \neq k$.

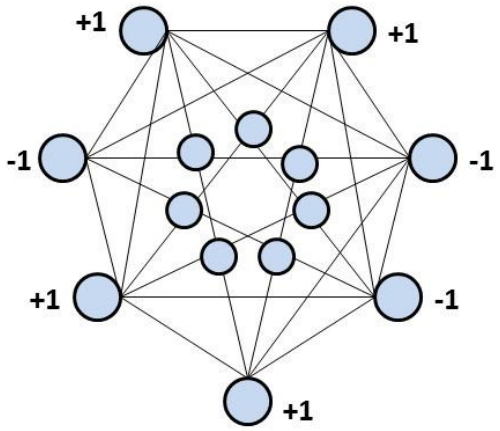


Figure 24. Inner and Outer Neurons in a Boltzman Learning model

3.4 Selected ANN models

The appropriate predictive mathematical model must offer accuracy and simplicity to learn from prior cases and easily be extensible to apply new data to make predictions about a patient's health condition. This prediction is possible by classification of a new patient into any one of possible disease categories. Since this dissertation uses ANN models for classification, it's important that close attention is given to accuracy of the model. The following four models were selected because each provides certain characteristic that make it appropriate for certain type of data and computation. It has also been established that these models are among the most accurate neural network models for classification. Below is a summary of advantages and disadvantages of each ANN method (Masters 1995):

1) PNN - Probabilistic Neural Networks are four layer networks. They classify data in a non-parametric method and are less sensitive to outlier data. It's been demonstrated that probabilistic neural networks using only four layers of input, pattern, summation and output perceptron can provide accurate and relatively faster classifications than the back-propagation neural networks (Principe, Euliano, Lefebvre 1999).

2) SVM – Support Vector Machine networks. SVM performs classification by constructing a two-layer network that defines a hyperplane that separates data into multiple classifications. The SVM is a non-probabilistic binary linear classifier. It takes a set of input data and determines which of possible classes the input is a member of.

3) GFN (Generalized Feed-forward) trained with LM – A feed-forward neural network consists of one or more layers of nodes where the information flows in only one

direction, forward from the input nodes and there are no cycles or loops in the network. In the multi-layer model, each node has direct connection to the nodes in the subsequent layer. The sum of products of the weights and the inputs are calculated in each node (Haykin 1999).

4) MLP trained with LM – Multi-layer perceptron, a method similar to gradient descent approach with variable step modification. Several variations of this model have been proposed, including the Levenberg-Marquardt model (Wilamowski & Chen, 1999) which is known to be among the most efficient algorithms.

The next section is a more in-depth mathematical review of the four Neural Network approaches used in this dissertation.

3.5 Probabilistic Neural Networks

Recall the classification problem from Figure 14 where the goal is to classify an unknown patient (shown by ?) into one of the two groups. The most straightforward method would be to check the distance from the nearest neighbor. But this method, while simple, has weaknesses. It can be misclassified into one group when in fact it belongs to another groups' cluster. The goal is to define a "sphere of influence" function to represent the spread of distance separating an unknown point from a training set point. Such a function would have a peak at zero distance from the training set point and taper off to zero as the distance increased. A proposed classifier would compute the sum of this function for all training set points of each population and classify the unknown into the population that has the greatest sum.

The following derivation is adapted from Sengupta (Sengupta 2009) and Zurada (Zurada 1997), improved and revised specific to this research. A mathematical construct that can help define such a function is the Gaussian function:

$$f(x) = ae^{-\frac{(x-x_i)^2}{2\sigma^2}} \tag{37}$$

Where a, x_i, σ are > 0, and a is the height of the curve's peak (or amplitude), x_i is the position of the center of the peak and σ is the width (or the spread) of the curve. (Hardy 2008) has illustrated a 2-dimensional graph for x_0, x_1 as shown in Figure 25, where the values of x_0, x_1 are set to origin (0,0). Coefficients a, and σ can take any positive values.

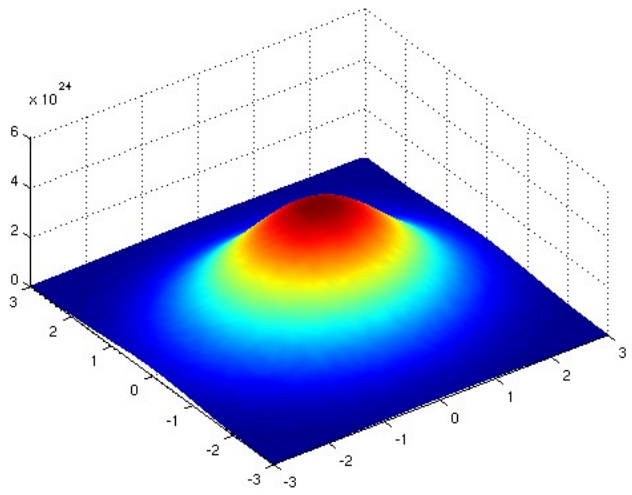


Figure 25. Graph of a 2-dimenstional Gaussian Function

Using the Gaussian function, (Parzen, 1962) and (Cacoullos, 1966) showed that one can compute the multivariate probability density function from a random sample. Essentially, Parzen's probability density function is a "sphere of influence" function that can be used as the classifier algorithm. The scaling parameter σ controls the width of the area of influence.

The idea behind PNN is that each training element represented by a Gaussian pattern unit, adds to the likelihood that nearby data has the same classification. To compute the classification of a data point, let's calculate the response for the point with every category and select the category that has the highest response. Each trained data point corresponds to pattern unit that is a Gaussian function with its peak centered on the parameter's location. The idea behind this classification approach is that for a new data point, we measure the average "distance" between the new point and all other points in the classification. The smaller the average "distance" of a point to other points results in a larger value of z as computed according to Equation (38). Therefore the point will belong to that classification where its z value is the highest.

Consider a simple Gaussian pattern equation that computes the result for a point (x_0, x_1) relative to already classified points (x_{0i}, x_{1i}):

$$z = f(x_0, x_1) = \sum_{i=1}^{n} e^{\frac{-\left((x_0 - x_{0i})^2 + (x_1 - x_{1i})^2\right)}{2\sigma^2}} \qquad (38)$$

Suppose one intends to predict the future state of a patient's Intra-Cranial Pressure (ICP). The goal is to classify the input data into one of 3 possible classes: Normal, Moderate or Critical. Each class is represented by f_N, f_M and f_C that are probability distribution functions for category N, M and C (Normal, Moderate and Critical). Then one can compute f_N, f_M and f_C as follows:

$$z_N = f_N(x_0, x_1) = \sum_{i=1}^{n} e^{\frac{-\left((x-x_{0Ni})^2 + (x-x_{1Ni})^2\right)}{2\sigma^2}} \tag{39}$$

$$z_M = f_M(x_0, x_1) = \sum_{i=1}^{n} e^{\frac{-\left((x-x_{0Mi})^2 + (x-x_{1Mi})^2\right)}{2\sigma^2}}$$

$$z_C = f_C(x_0, x_1) = \sum_{i=1}^{n} e^{\frac{-\left((x-x_{0Ci})^2 + (x-x_{1Ci})^2\right)}{2\sigma^2}}$$

Here the data point to be classified is x and all the other points that belong to other classifications are referred to by N_i, M_i and C_i. There are many activation functions possible but a common non-linear operation is the following:

$$e^{\frac{-(W_i-X)^t(W_i-X)}{2\sigma^2}} \tag{40}$$

If X and W_i are normalized to unit length, it's been demonstrated (Zaruda 1997) that the non-linear operation above can be replaced by (41). The derivation follows by multiplying the numerator terms that results in:

$$-w_i^2 + 2\, w_i X - X^2 = -2 + 2w_i X = 2(w_i X - 1)$$

Since the terms w_i^2 and X^2 are normalized to unity, they are replaced by 1. Substituting Z_i for $W_i\, X$ in the above term in the numerator, one can obtain the following activation function:

$$e^{\frac{(Z_i-1)}{\sigma^2}} \tag{41}$$

So a simple algorithm to identify classification of a new data set can be described as:

1. Input layer: Normalize X and W_i to unit length
2. Pattern layer: Compute the dot product of input X and weights of X, W_i
3. Summation layer: Compute f_N, f_M and f_C
4. Output (or decision) layer: Select output with highest response value (from N, M or C clusters of neurons)

The final classification is determined by a classifier function *C* that selects the largest of f_N, f_M and f_C values:

$Prediction_{(t+t1)} = C\ (f_N, f_M, f_C) = max(f_N, f_M, f_C)$ (42)

An Example in Appendix B illustrates how PNN can be applied to a simple classification problem.

3.6 Support Vector Machine (SVM) Networks

Support Vector Machines (SVM) are among supervised training models that analyze data for multiple classification and regression analysis. The SVM is a no-probabilistic binary linear classifier. It takes a set of input data and determines which of possible classes the input is a member of. SVM constructs a set of hyperplanes between data elements to classify them. A good separation is the mark of a generalizable model and is achieved by the hyperplane that has the largest distance to the nearest training data element in any class. The hyperplane is mathematically defined as the set of data elements whose inner product with a vector in that space is constant. Margin is the distance between the optimal hyperplane and a vector that runs close to it. The following derivation is adapted from Sengupta (Sengupta 2009) and Haykin (Haykin 1998), improved and revised specific to this research.

The most optimum solution can be found by gaining the biggest possible margin. The optimal hyperplane must satisfy:

$$\frac{y_k F(x_k)}{||w||} \geq \tau\ ,\ k = 1, 2, \dots, n \quad (43)$$

where τ is the margin and can be visualized as a band that separates the nears points from the hyperplane that separates them into two categories Note that F(x) is defined by:

$$F(x) = w^T x + b \quad (44)$$

One can map the data points to a very high-dimensional space, then the algorithm finds a hyperplane in this space with the largest margin separating classes of data. The feature space is usually defined as a non-linear product of base functions $\varphi_i(x)$, defined in the input space. Then function *F(x)* becomes:

$$F(x) = \sum_{i=1}^{n} a_i K(x_i, x) + b \quad (45)$$

where $K(x_i, x)$ is the inner product kernel of base functions $\varphi_i(x), j = 1, 2, \ldots, m$. The cross products in the larger space are defined by a kernel function *K(x, y)* that best fits the problem, such that:

$$\sum_i a_i K(x_i, x) = constant \tag{46}$$

It can observed that *K(x,y)* becomes small as *y* grows further from x. The inner product K(.) can have many possible kernels, one of the most commonly used is based on the Gaussian:

$$K(x_i, x) = e^{(-\frac{\|x - x_i\|^2}{2\sigma^2})} \tag{47}$$

where $\sigma > 0$, and sigmoid kernels

$$K(x_i, x) = \tanh(\theta < x_i, \ x > + \vartheta) \quad \text{such that } \theta > 0, \ and \ x > + \vartheta. \tag{48}$$

3.7 General Feed-forward Neural Network

A feed-forward neural network consists of one or more layers of nodes where the information flows in only one direction, forward from the input nodes and there are no cycles or loops in the network. The simplest networks have single layer that feeds input data to the output layer via a series of weights. One of the popular training methods is the Gradient Descent algorithm. The following is a mathematical development of the Gradient descent method, adapted from Sengupta (Sengupta 2009) and Zurada (Zurada 1997) and improved for this research.

For a given neural network with *n=1,...,N* layers, one can compute the error for each node. By definition the error at time snapshot *n,* for node *j* computed by:

$$e_j(n) = d_j(n) - y_j(n) \tag{49}$$

And for computing total error of a network, recall Equation (21) provides that:

$$E(n) = \frac{1}{2} \sum_j e_j{}^2(n) \qquad \text{for } j=1 \text{ to } m. \tag{50}$$

Then one can compute average error of a network by:

$$E(N)_{Avg} = \frac{1}{N} \sum_{n=1}^{N} E(n) \tag{51}$$

Let's use the traditional Pitts-McCullough equation to compute:

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n) y_i(n) \ , \tag{52}$$

where $v_j(n)$ is the input to activation function for the neuron j. The term $v_j(n)$ can be perceived as the induced local field while $y_j(n)$ can be thought of as the output from the previous layer, shown by:

$$y_j(n) = \varphi\big(v_j(n)\big) \tag{53}$$

Given the prior introduction, it's possible to start the mathematical derivation of back-propagation method. First it's easy to calculate the partial derivative of *E(n)* and apply the chain rule to obtain the following:

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = \frac{\partial E(n)}{\partial e_j(n)} \cdot \frac{\partial e_j(n)}{\partial y_j(n)} \cdot \frac{\partial y_j(n)}{\partial v_j(n)} \cdot \frac{\partial v_j(n)}{\partial w_{ji}(n)} \tag{54}$$

Where Δw_{ji} is applied to w_{ji}. Let's apply the derivatives to each component above:

$\frac{\partial E(n)}{\partial e_j(n)} = e_j(n)$

$\frac{\partial e_j(n)}{\partial y_j(n)} = -1$

$\frac{\partial y_j(n)}{\partial v_j(n)} = \varphi'\big(v_j(n)\big)$, and one can take derivative of $\varphi\big(v_j(n)\big)$ when the exact function is known.

$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n)$

Now it's possible to write the result of substitutions as:

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = -e_j(n)\, \varphi'\big(v_j(n)\big)\, y_i(n) \tag{55}$$

When adjustment of $\Delta w_{ji}(n)$ (namely the correction to $w_{ji}(n)$ is applied to $w_{ji}(n)$ the following relationship can be obtained:

$$\Delta w_{ji}(n) = -\eta \frac{\partial E(n)}{\partial w_{ji}(n)} = \eta\, e_j(n)\, \varphi'\big(v_j(n)\big)\, y_i(n) \tag{56}$$

This equation is a reminder that $\Delta w_{ji}(n)$ is proportional to $\frac{\partial E(n)}{\partial w_{ji}(n)}$ at a rate of proportionality η, negative to the direction of the gradient. Part of the term in (55) can be described as error multiplied by activation function. This term can be shown as:

$$\partial_j(n) = -\frac{\partial E(n)}{\partial v_j(n)} = -\frac{\partial E(n)}{\partial e_j(n)} \cdot \frac{\partial e_j(n)}{\partial y_j(n)} \cdot \frac{\partial y_j(n)}{\partial v_j(n)} \tag{57}$$

$$\partial_j(n) = e_j(n)\, \varphi'\big(v_j(n)\big) \tag{58}$$

The term $\partial_j(n)$ is the derivative of error with respect to the activation function $v_j(n)$. This is the gradient for neuron *j* and is local to neuron *j*. This is also known the local gradient. Now one can rewrite the equation (56) as:

$$\Delta w_{ji}(n) = \eta\, \partial_j(n)\, y_i(n) \quad \text{where } y_i(n) \text{ is the input to neuron } j. \tag{59}$$

Since the values for η and $y_i(n)$ are known it's possible to compute $\partial_j(n)$. Two cases are possible:

Case 1) Neuron *j* belongs to the output layer, hence $\partial_j(n)$ can be calculated from equation (54).

Case 2) Neuron *j* belongs to a hidden layer, thus $\partial_j(n)$ must be computed differently.

A fundamental condition for back-propagation is that one can compute $\partial_j(n)$, namely the derivative of the activation function is possible. Let's examine this approach graphically to illustrate the hidden layer and output layer computations as shown in Figure 26. The signal flow graph in Figure 26 shows *j* is the hidden layer and *k* as output layer neurons.

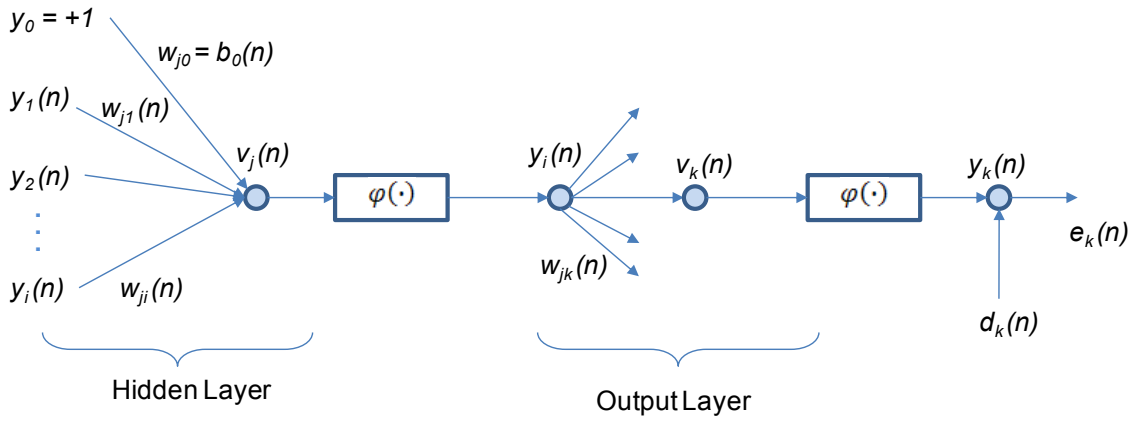


Figure 26: Depiction of hidden layer *j* and output layer *k* neurons

One can compute $\partial_j(n)$ by the following derivations. It's known that:

$$\partial_j(n) = -\frac{\partial E(n)}{\partial v_j(n)} = -\frac{\partial E(n)}{\partial y_j(n)} . \varphi'\big(v_j(n)\big) \tag{60}$$

Also recall from equation (50) that:

$E(n) = \frac{1}{2} \sum_k {e_k}^2(n)$, where k is the set of output layer neurons. It can be shown that:

$$\frac{\partial E(n)}{\partial y_j(n)} = \sum_k e_k(n) \frac{\partial e_k(n)}{\partial y_j(n)} = \sum_k e_k(n) \frac{\partial e_k(n)}{\partial v_k(n)} . \frac{\partial v_k(n)}{\partial y_j(n)} \tag{61}$$

$$\text{Since } e_j(n) = d_j(n) - y_j(n) = d_j(n) - \varphi(v_j(n)) \tag{62}$$

Let's take derivatives of the left hand side:

$$\frac{\partial e_k(n)}{\partial v_k(n)} = -\varphi'\big(v_k(n)\big) \tag{63}$$

For neuron k, it's possible to write:

$$v_k(n) = \sum_{j=0}^{m} w_{kj}(n) y_j(n) \tag{64}$$

Consequently by taking derivate of (64), it results in the following:

$$\frac{\partial v_k(n)}{\partial y_i(n)} = w_{kj}(n) \tag{65}$$

$$\frac{\partial E(n)}{\partial y_j(n)} = -\sum_k e_k(n)\ \varphi'\big(v_k(n)\big)\, w_{kj}(n) = -\sum_k \partial_j(n)\, w_{kj}(n) \tag{66}$$

From equation (60) and (66) one can conclude that:

$$\partial_j(n) = \varphi_j'\big(v_j(n)\big) . \sum_k \partial_k(n)\, w_{kj}(n) \tag{67}$$

This equation is significant as it shows that the local gradient of neuron j (hidden neurons) depends on the local gradient of output neuron k. The summation is the weighted sum of all the output gradients. Let's assume M is the number of output neurons. Consider multiplying each error term $e_j(n)$ by the derivative of the corresponding activation function $\varphi_k(v_k(n))$, namely by $\varphi_k'(v_k(n))$. This is the basis of back-propagation as depicted in Figure 27.

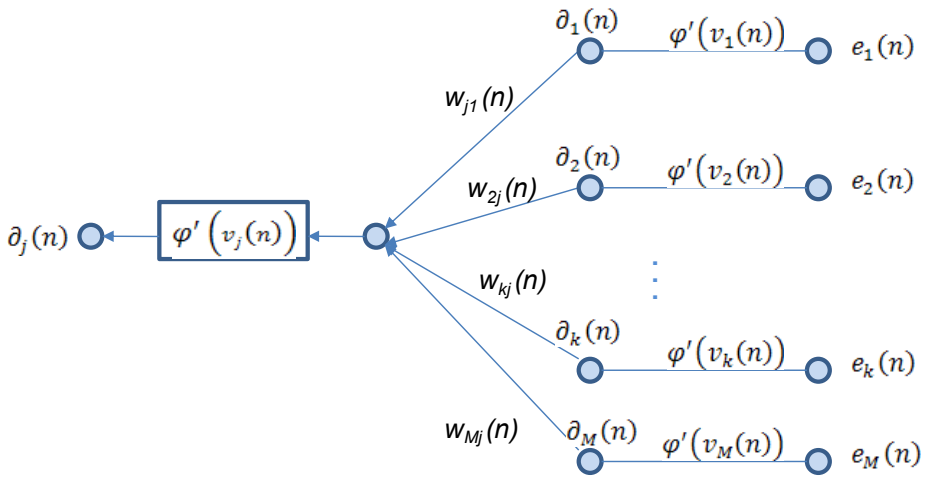


Figure 27: Backward propagation - computing $\partial_j(n)$ from errors $e_k(n)$ in forward step

In forward pass, the inputs propagate forward from 1st layer to the 2nd layer and so on to eventually to the output layer. The error terms are computed and then the backward

propagation begins. In backward propagation, the algorithm starts with the error term. Then it calculates the local gradients and propagate back and adjust the synaptic weights.

Activation function can be any of the three types mentioned previously, such as the sigmoid function, logistic function or a *tanh()* function. Let's compute the derivative of activation function for the case of the logistic function:

$$\varphi_j\left(v_j(n)\right) = \frac{1}{1+\mathrm{e}^{\left(-av_j(n)\right)}} \equiv y_j(n) \tag{68}$$

It's given that a >0, and $-\infty < v_j(n) < \infty$. It's possible to compute the derivative of the activation function as follows:

$$\varphi_j'\left(v_j(n)\right) = \frac{a\,.\,e^{(-av_j(n))}}{[1+e^{(-av_j(n))}]^2} = a\,.\,y_j(n)\left[1-y_j(n)\right] \tag{69}$$

In other words one can write the above in the following fashion:

$$\varphi_j'\left(v_j(n)\right) = \left[1-y_j(n)\right].(a)\,.(y_j(n)) \tag{70}$$

Now one can use $y_j(n)$ for computing $\partial_j(n)$ by:

$$\partial_j(n) = \varphi_j'\left(v_j(n)\right)\,.\sum_k \partial_k(n)\,w_{kj}(n) =$$

$$a\,.\,y_j(n)\left[1-y_j(n)\right]\sum_k \partial_k(n)\,w_{kj}(n) \tag{71}$$

The range of $y_j(n)$ is [0,1]. The value of $\varphi_j'\left(v_j(n)\right)$ is maximum when $y_j = 0.50$, and $\varphi_j'\left(v_j(n)\right)$ is equal to zero when $y_j = 1.0$ or zero. This fact guides our choice of proper values for Δw_{kj}. Similarly one could have taken derivative of $tanh(x)$, to compute Δw_{kj}. A detailed description of this algorithm appears in Appendix C.

3.8 MLP with Levenberg-Marquardt (LM) Algorithm

Feed-forward MLP with LM is a feed-forward neural network that consists of one or more layers of nodes where the information flows in only one direction, forward from the input nodes and there are no cycles or loops in the network (Sengupta 2009, Masters 1995). The simplest networks have single layer that feeds input data to the output layer via a series of weights. In the multi-layer perceptron model (MLP), each node has direct connection to the nodes in the subsequent layer. The sum of products of the weights and the inputs are calculated in each node (Haykin 1999). If the value of the result is above a certain threshold, the neuron fires with the activated value (typically 1), otherwise, it fires the deactivated value

(typically -1). Several variations of this model and training methods have been proposed, including the backward propagation algorithm and Levenberg-Marquardt (LM) method which is considered as one of the more computationally efficient algorithms. The following derivation is adapted from Sengupta (Sengupta 2009), Haykin (Haykin 1998) and Masters (Masters 1995), revised and improved for this research.

The training of MLP occurs in two stages: in a forward phase the weights of the network are fixed and the input data is propagated through the network. The forward phase completes its computation with an error signal. The error term was defined by Equation (49) and can be written specifically to can be defined as

$$e_{kp} = d_{kp} - y_{kp},\ k = 1, \dots, K,\ p = 1, \dots, P \tag{72}$$

where d_{kp} is the desired response and $y_{kp}$is the actual output produced by the network response to the input x_{ip}. d_{kp} is the desired value of the k^{th} output and the P^{th} layer. Y_{kp} is the actual value of the k^{th} output and P^{th} pattern. The parameter K is the number of network outputs, P is the number of patterns and N is the number of weights.

The backward phase the error e_{kp} is propagated through the network going backward and the free weights are adjusted to minimize error e_{kp}. In the LM algorithm, the performance index $F(W)$ is to be optimized:

$$F(W) = \sum_{p=1}^{P}\left[\sum_{k=1}^{K}(d_{kp} - y_{kp})^2\right] \tag{73}$$

Where W = $[w_1\ w_2\ \dots\ w_N]^T$ is the set of all weights for the network. The equation can be written as:

$$F(W) = E^T E$$

Where $E = [e_{11}\ \dots.\ e_{K1}\ e_{12}\ \dots\ e_{K2}\ \dots\ e_{1P}\ \dots\ e_{KP}]^T$ (74)

The error term E, is the cumulative error vector for all patterns. Let's assume that the amount of change to each weight is shown by .Using the Jacobian matrix one can compute the amount of change that be applied to weights in the backward propagation. By definition, a Jacobian is the derivative of one vector with respect to another vector. From the equation above, the Jacobian matrix can be defined as:

$$J = \begin{bmatrix} \frac{\partial e_{11}}{\partial w_1} & \frac{\partial e_{11}}{\partial w_2} & \cdots & \frac{\partial e_{11}}{\partial w_N} \\ \frac{\partial e_{21}}{\partial w_1} & \frac{\partial e_{21}}{\partial w_2} & \cdots & \frac{\partial e_{21}}{\partial w_N} \\ \vdots & \vdots & & \vdots \\ \frac{\partial e_{K1}}{\partial w_1} & \frac{\partial e_{K1}}{\partial w_1} & & \frac{\partial e_{K1}}{\partial w_1} \\ \vdots & \vdots & & \vdots \\ . & . & & . \\ \frac{\partial e_{1P}}{\partial w_1} & \frac{\partial e_{1P}}{\partial w_2} & \cdots & \frac{\partial e_{1P}}{\partial w_N} \\ \frac{\partial e_{2P}}{\partial w_1} & \frac{\partial e_{2P}}{\partial w_2} & & \frac{\partial e_{2P}}{\partial w_N} \\ \vdots & \vdots & & \vdots \\ \frac{\partial e_{KP}}{\partial w_1} & \frac{\partial e_{KP}}{\partial w_2} & & \frac{\partial e_{KP}}{\partial w_N} \end{bmatrix} \tag{75}$$

Using the Newton-Raphson method, one can compute the change in weight by applying a dampened measure of error terms. It's possible to derive the Levenberg-Marquardt algorithm as follows. Recall that the output is a function of inputs x_i and weights W. This can be stated by writing:

$$y_i = y_i\,(x_i, W)$$

In other words, y_i depends on input x_i and weights W. The goal of backward propagation is to adjust the weights W by $J_i\delta$ where δ is the amount of adjustment and J_i is the i^{th} row of the Jacobian matrix. Then it's possible to write this expression as:

$$y_i(x_i, W + \delta) \cong y_i\,(x_i, W) + J_i\delta$$

The goal of computation is to minimize the objective function in Equation (73):

Minimize $E(W) = \sum_p \sum_k (d_i - y_i)^2$

When adjusted by δ, the minimization function can be written as:

$$E(W + \delta) = \sum_p \sum_k (d_i - y_i(x_i, W + \delta))^2 \cong \sum_p \sum_k (d_i - y_i - J_i\delta)^2$$

Next the error term can be written as:

$$E(W + \delta) \cong \left\| \bar{y} - \bar{d} - \boldsymbol{J}\delta \right\|^2 \tag{76}$$

To minimize this function, one takes the derivative and sets it to zero. The derivative of the above function set to zero becomes:

$-2\boldsymbol{J}^T\left(\bar{d} - \bar{y} - \boldsymbol{J}\delta\right) = 0$, namely:

$$\boldsymbol{J}^T\left(\bar{y} - \bar{d}\right) = \boldsymbol{J}^T\boldsymbol{J}\delta$$

The rate of change can be tempered by a scalar multiple shown by parameter η. Equation (76) can be written as:

$\boldsymbol{J}^T(\bar{y} - \bar{d}) = (\boldsymbol{J}^T\boldsymbol{J} + \eta\boldsymbol{I})\delta$, or as:

$\boldsymbol{J}^T\boldsymbol{E} = (\boldsymbol{J}^T\boldsymbol{J} + \eta\boldsymbol{I})\delta$

where *I* is the identity matrix. It can be seen that the rate of change in weights can be computed by:

$$\delta = \frac{\boldsymbol{J}^T\boldsymbol{E}}{(\boldsymbol{J}^T\boldsymbol{J}+\eta\boldsymbol{I})}$$

It can be easily seen that the weights for the next iteration can be computed using the following equation:

$$W_{t+1} = W_t - (\boldsymbol{J}_t^T\boldsymbol{J}_t + \eta_t\boldsymbol{I})^{-1}\boldsymbol{J}_t^T\boldsymbol{E}_t \quad (77)$$

Where $\boldsymbol{J}$ is the Jacobian (a matrix of first order derivatives) of *m* input errors with respect to *n* weights of the neural network, $\boldsymbol{I}$ is the identity matrix and η_t is the learning parameter.

3.9 Use of Ensembles to Improve Accuracy

The traditional data-driven prediction methods have constructed multiple models and selected the one with the best performance, discarding all the others. This approach has some disadvantages: 1) the effort of constructing several models is wasted, 2) the selected model may not consistently perform most accurately or be robust on all types of data and with diverse types of disease predictions, 3) the selected algorithm may not be able to sustain or perform consistently on training data types as changes to data types occur over time. To overcome these disadvantages, this dissertation proposes a multi-model ensemble (also known as committee of models) approach which combines multiple algorithms with a weighted sum formulation. This research considered five different ensemble schemes and compared their performance on the DVT case study data set. These schemes include a voting formula, two accuracy-based weighting schemes, a diversity-based weighting and optimization-based weighting. The goal of constructing ensembles is to identify the weights of each algorithm such that it improves data-driven prognostics performance.

The case study employed in this dissertation demonstrates that the ensemble approach provides a more accurate prediction than a single algorithm. Given a number of neural networks to select from, the goal is to select a weighted sum of these models' output that provides the most accurate classification. An oracle program can be defined to select the

most accurate algorithm from a set of five ensembles (or committee) of algorithms provided by the four ANN models. An oracle is defined as an overseer which selects the most appropriate answer amongst a set of options. An oracle, is a program that selects a prediction from among a number of ensembles or models that meet a desired level of accuracy or predictive characteristic of a disease.

Since one model performs better in predicting true positives and another better at predicting the true negatives, this dissertation proposes the oracle program to combine the predictions from models in a way that the model with higher accuracy is assigned a higher weight and the worst model still contributes to the prediction but at a smaller weight. This way, the oracle can improve the classification accuracy, sensitivity and specificity by combining the best classification characteristics from different models.

Given that there are many neural networks to select from, the goal is to select the most accurate model, or ensemble of models for prediction of a particular disease. Let's as an example suppose that the prognostics engine trains four different algorithms to make a prediction for DVT. It can then build five different ensembles with different weighted sum from the four models' output. The prognostics engine trains another four models to predict a different disease and builds another set of five ensembles. For each disease, a different set of ensembles are constructed. Then the prognostics engine uses an oracle, an overseer program that selects the most accurate ensemble for each disease prediction.

In prior research, a meta-classifier is used to compose a prediction from the four models based on a linear combination of the characteristics that are desired in the final classification. A meta-classifier is a computational tool that integrates in some principled fashion the separately trained classifiers to boost overall predictive accuracy (Prodormidis, Chan, Stolfo 2000). In this research, this meta-classifier is called the oracle program. The oracle program can be set up to enhance any of the four desired accuracy characteristics: either directly improve True Positives (TP), True Negatives (TN), False Positives (FP) or False Negatives (FN), or to improve other accuracy measurements such as sensitivity, specificity or Youden's *J* index.

In this research, a positive case is a patient with illness and a TP classification means the algorithm has correctly classified the ill patient as sick. Then FP represents Type I error (α) and FN represents Type II error (β) as shown in Figure 28.

		Truth: Sick	Truth: Healthy
Test	Sick	TP (1- β)	FP α
	Healthy	FN β	TN (1- α)

Figure 28. Truth Table indicating Type I and Type II errors

It's important to define the objectives for the oracle program. The practitioner must define the selection criteria for the oracle program; is the oracle to select a model or ensemble of models that reduce Type I (FP) error, or Type II (FN) error, or both errors; or as it will be presented later, a combination of accuracy measures.

While in most situations, the criteria calls for lower Type II error, in certain situations where the population is large and the cost of treatment is high, the criteria would include reducing Type I error as well.

The oracle program can be set up to provide a meta-classification based on a linear combination of these characteristics. The goal of the ensemble is to produce a synthesized classification result based on weighted sum of ANN models as:

$$\hat{p} = \sum_{j=1}^{K}(w_j p_j) \tag{78}$$

The assumption in this dissertation is that the ensemble consists of K models, and p_j is the prediction of model j. The result of the ensemble's prediction is represented by $\hat{p}$.

3.9.1 Vote-based Schema

Voting approach selects the final results based on majority vote on a specific accuracy dimension or result category. For example, if the goal is to minimize FN, then one must consider a voting schema that produces the lowest FN among the four models on a given new case data set. The voting schema works as follows: Run all four models on the new data set. Select the lowest FN count, or select a weight sum of models that produce the lowest FN+FP count.

A hybrid schema can work by defining a relative preference ratio. The assumption is that models with more correct prediction are preferred over the other models. The relative preference of a model is determined by the number of correct predictions minus incorrect predictions. This schema was used for Ensemble #1. The mathematical representation of

schema for Ensemble#1 is shown in (79) and (80). This ensemble assumes *n* data cases and *K* models in the ensemble:

$$model\ j\ score = \sum_{i=1}^{n}(correct\ predictions - incorrect\ predictions) \quad (79)$$

$$w_j = \frac{model\ j\ score}{\sum_{i=1}^{K} model\ i\ score} \quad (80)$$

As shown in this equation, the intent is to find the weights according to the ratio of preference for a model that provides higher accuracy (lower FN count).

3.9.2 Accuracy-based Ensemble Schema

Accuracy-based ensemble seeks to find a weighted sum of algorithms that minimize error. For example, if the intent is to find the lowest FN, one can define the error for each validation case as follows. Let's assume that T_i is the truth of a particular validation case, and p_{ij} is the prediction rendered by algorithm *j* for case *i*. Also assume that there are a total of *K* different algorithms. Ensemble #2, attempts to increase the number of TP count by defining the number of errors for each model *j*:

Then one can define the total error count by:

$$\varepsilon_j = \sum_{i=1}^{n} \varepsilon(p_{ij}\ ,\ T_i) \quad (81)$$

where $\varepsilon(p_{ij}\ ,\ T_i)$ equals to 1 if p_{ij} does not match the truth T_i, or equals to zero if prediction matches the truth.

Then one can define the weight of algorithm *j* for ensemble #1with the goal of increasing the quantity of TP. Assuming that the total number of TP from a model *j* is shown by TP_j, it's possible to define the weights by:

$$w_j = \frac{(TP_j)}{\sum_{j=1}^{m}(TP_j)} \quad (82)$$

For ensemble#3, one can define the weight of algorithm j with the goal of reducing the quantity of FN error:

$$w_j = \frac{(\varepsilon_j)^{-1}}{\sum_{j=1}^{K}(\varepsilon_j)^{-1}} \quad (83)$$

In this research, two ensembles, Ensemble#2 and Ensemble#3 used accuracy-based scheme to determine the weights. The results are discussed and compared in Chapter 5.

3.9.3 Diversity-based Schema

The accuracy-based weighted sum formulation exclusively relies on accuracy to compute the weights. However, one can argue that accuracy is not the only factor that affects ensemble performance. The diversity-based schema measures the extent to which the predictions by one model are distinguishable from predictions by other models. The diversity-based schema increases the robustness performance of the ensemble. Said differently, this ensemble assigns higher weight to the model with higher prediction diversity because it offers higher ensemble robustness. Let's assume that *n* data sets are employed to train and validate all *k* models. One can compute a prediction error term u_j to correspond to uniqueness of the model *j*:

$$\theta_j = (p_{j1} - T_1, \dots, p_{jn} - T_n) \tag{84}$$

Given k algorithms, it's possible to define the error vector of $\theta_1, \theta_2, \dots, \theta_K$. The diversity of the *j*th model can be computed as the sum of Euclidean distances between the vector θ_j and all other error vectors, defined by:

$$D_j = \sum_{i=1;j\neq i}^{K} \|\theta_j - \theta_i\| \tag{85}$$

The prediction diversity determines how a model's result is distinguishable from those of other models. Let's compute a normalized weight w_j for the *j*th model in the ensemble by:

$$w_j = \frac{D_j}{\sum_{i=1}^{k} D_i} \tag{86}$$

Ensemble#4 uses diversity-based schema.

3.9.4 Optimization-based Schema

One proposal in this dissertation is to define the oracle (meta-classifier) as an optimization model. The optimization-based schema can take into account both the accuracy-based as well as the diversity-based weighting scheme to improve accuracy and robustness. This method is adapted from Hu, et al. (Hu, Youn, Wang 2010). In optimization-based schema one can write Eq. (78) as:

$$minimize\ \varepsilon\left(\sum_{j=1}^{K}(w_j\, p_{ij}), T_i\right) \tag{87}$$

$$\sum_{j=1}^{K} w_j = 1.\text{, and } \ w_j \geq 0\ for\ all\ j = 1, \dots K \tag{88}$$

Where w_j is the weight of model p_j and T_i is the expected result for the i^{th} data set. The objective function attempts to minimize the difference between the expected result and the weighted sum of each model's result. Ensemble#5 uses optimization-based schema. The weights w_1, w_2, w_3 and w_4 corresponding to each of the four ANN models were determined using the optimization model in Excel Solver as is illustrated in the upcoming section 5.4 by Figure 40.

Chapter 4

4 CASE STUDY, DATA, SOFTWARE, ANALYSIS

4.1 Data requirements for Learning

Researchers, who have studied learning from data, have classified two approaches to learning: programmed and concept attainment. Programmed learning is applicable when the researcher knows the underlying causal relationships in the system and the data. The concept attainment is an adaptive learning approach where a system learns from *a priori* set of examples and thus retains concepts from prior data sets, in other words it learns by classifying patterns in the input data. This research applies the concept attainment type of learning using ANNs as the learning models. A concept can be described as a mapping between an input data set and a clinical outcome.

Some interesting research questions arise that merit a full investigation elsewhere, but are discussed briefly here:

A) How much data is adequate for learning to predict a given disease?

B) How much of the error in prediction can be attributed to noise in data versus the model that captures the concept or due to the change in concept?

C) What pre-processing methods are applied to measured data to prepare data for learning algorithms?

The answers are briefly discussed as follows:

4.1.1 How much data is adequate for learning to predict a given disease?

In order to train a Prognostics engine adequately, the ideal data set must consist of adequate number of both positive and negative cases of disease (Principe, Euliano, Lefebvre 1999). From experience, some basic rules of thumb have been developed for determining the amount of data needed for proper training of ANN models. The rules have been developed as guidelines by ANN experts (Principe 2011).

1. A data set must have a minimum of 5 times as many exemplars (data sets) as the number of weights in the ANN model. For example, in the DVT model that includes 20 input variables, and 4 layers of network, there can be as many as 60 weights. Thus the minimum number of input data should be somewhere around 300 rows of data.

2. The data set must have approximately 50 times the number of exemplars (data sets) than features. A feature is the number of columns. In the DVT model that includes 20 columns, it's recommended to need a minimum of 1000 rows of data (50 times more rows than columns)
3. In classification models, the number of rows in the smallest class should be preferably 5 times the number of input columns. For example, in the DVT model that has two classifications (disease or no-disease), the number of data rows in the smaller class must be over 100 rows.

In order to properly train a model, it's recommended to include equal number of positive and negative cases to have a balanced network. Let's suppose the goal is to train a model with 400 negative cases and only 50 positive cases, the ANN model naturally works to minimize errors. As a result, the model gets trained by the input data such that minimizing errors for the 400 cases overshadows the minimizing errors for the 50 positive cases. So, it's recommended to randomly select 50 cases from negative and 50 cases from negative sample and train on this set.

4.1.2 How much of the error in prediction can be attributed to noise in data versus the model that captures the concept or due to the change in concept?

Data noise can be caused by incorrect measurements of patient physiological data, disruptions to measurements (missing data) and or incorrect *a priori* classification or units of measure. Concept changes on the other hand can be attributed to changes to how patients, medications and treatment protocols and practices change over time. For example, certain diseases develop resistance to certain drugs over time so those drugs are not as effective, or the procedures for a treatment change that make it either more or less effective, but it's prescribed by same name.

By conducting a comparison of results of four different ANN models, the difference in accuracy among the models can give us some clues to the amount of error that can be attributed to the model.

4.1.3 What pre-processing tools are used to prepare measured data for learning algorithms?

In most neural network models, all data in each column is normalized such that an amplitude and an offset is applied to the data of each column. It's important to study the data

before training a model. If the input data ranges are vastly different (for example, one data column has a range of 1000 and another has a range of 1), then the errors from multiplying weights by inputs that have a large range will overtake the smaller data range.

4.2 Input Data Pre-Processing

The goal of pre-processing is to bring the range of input data values within a computationally acceptable range for the ANN computations. ANNS train faster and perform better if their data is pre-processed. The same pre-processing must be applied to test data. Data scaling is an important pre-processing step before training ANNs. Data scaling equalizes the importance of input variables at the input layer. For example, if one input variables ranges between 1 and 10,000 and another ranges between 0.001 and 0.1, the network should be able to use proper initial weights, namely small weights for the first variable but larger weights for the second variable. But, data scaling make the choice of initial weights for ANNs easier so they can train faster.

At the input layer, several methods are available to pre-process the measured data for ANN models. These methods include:

1. **Moving average:** Computes the moving average of a column using the chosen window length.
2. **Difference:** Computes the difference or percent difference along a column of data from the mean of the column.
3. **Clip data:** Clip data to a given max value or min value
4. **Log of data:** Takes logarithm of each data item
5. **Mean and Variance:** Normalize the data by fixing the mean to zero and use variance of from the mean to scale the data

A commonly used scaling method normalizes input data to unit length. Normalizing to unit length implies that the sum of squares of values in a given data set are equal to 1. To normalize each data value, the following steps are taken:

1. Square all data values in a given data set
2. Sum the squares. Take square root of the sum of squares
3. Divide each data item in the data set by this square root of sum of squares. The result of each division is a normalized data item.

Other pre-processing and coding techniques are necessary for categorical data. For example, let's assume a particular patient data is recorded as a categorical input variable such that it takes values of Very Low, Low, Medium, High and Very High. This variable should not be coded in numeric values of 0.0, 0.25, 0.50, 0.75, 1.00, as this would create incorrect interpretations by the ANN model. For example, it would incorrectly imply that a High category is exactly 3 times more than a Low value. Such input variables are coded and transformed into binary inputs. In this example, the categorical values would be mapped as shown in Table 5:

Table 5. Coding example of categorical data

Category	Binary Input
Very Low	0 0 0 0
Low	1 0 0 0
Medium	1 1 0 0
High	1 1 1 0
Very High	1 1 1 1

Additional data pre-processing occur at each layer of ANN models. After each layer, the range of normalized values is determined by the range of outputs of the nonlinear transfer functions (activation function) used by the model. For example, if a *Tanh* axon is used, the data output is normalized to a range between -1 and +1. If the data for the column is already in the desired range, then the amplitude will be 1 and the offset will be 0, so that the normalized data will be the same as the original data. There are other options that clean and randomize data depending on what is intended for the initial data processing. Some example methods are:

- **Randomize rows:** Randomly re-arrange rows of data.
- **Clean data:** replace missing or corrupted data with an average of the column, ore the most recurring data or the nearest value in the column.

The measured data may come from a number of acquisition devices and data bases. Each training set is represented by input data vector **X** and a prognostic disease ***Z,*** so one can represent each training set as $\{(\mathbf{x}_1,z_1),\ldots, (\mathbf{x}_n, z_n)\}$ for some unknown function $\boldsymbol{Z = P(X)}$. Each x_i is a set of attributes (feature) vector $\boldsymbol{X}$ of the form $\{x_{i1}, x_{i2}, \ldots, x_{ik}\}$, and each z_i represents the prognostic disease label associated with each vector ($\boldsymbol{Z_i} \in \{z_1 , z_2 , \ldots, z_n\}$).

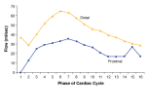
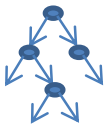
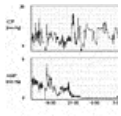
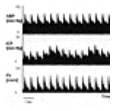
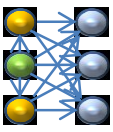
Our task is to compute a classifier or model $\widehat{\boldsymbol{P}}$ that approximates $\boldsymbol{P}$ and correctly labels any feature vector drawn from the same vector source as the training set.

Once the data is collected into a single data base, called Memory, it can be used to train several classifiers that can classify a new data set into a number of disease types. This research trained four classifier models using N-leave-out method. In this approach, each model is trained on a portion of the data and leave *N* data sets out of training to be used for testing. This process continues until all data sets have been used training and testing. This training approach reduces the effect of data bias.

4.3 Data Acquisition for ANN models

ANN models require real time data acquisition from several sources including patient vital sign monitors, lab results, waveform data, data acquired from clinical instruments, medical record and other diagnostic tests. The challenge with managing a diverse set of data can be described in the taxonomy shown in Table 6. As explained in this table, analysis must consider all types of data in real time as each data source is updated. In this research 1,073 cases were analyzed with are adequate for proper training of ANN models.

Table 6. Clinical Data Types Are Diverse

Nominal	Ordinal	Ratio	Fuzzy - Range	Image	Signal	Graph	Clinical Charts
Male	High	34.22					
Female	Low	?					
. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .
Male	High	-5.70					

The input file for each ANN model must accommodate the necessary medical data types from the taxonomy above. The challenge is handling images, missing data elements (when there is interruption to measuring clinical information) and various types of clinical charts. The goal is to extract the relevant medical data from images and charts in form of

discrete data elements. Signal and Range data can be represented by time-series data and data matrix format.

4.4 Case Study: Application of ANN to DVT/PE Data

Patients in acute care can develop a multitude of complications ranging from pulmonary, respiratory and digestive to infections. The goal of this research is to collect such data in 1 minute intervals and study the changes in data to predict patient's health condition using ANN models developed for each type of complication. The challenge is to run the models in real time, once every minute as new data becomes available. The choice of time window can vary from several minutes to several hours. The ANN models dedicated to each human physiological system predict complications in each category. The physicians can apply the appropriate treatment before such complications occur or escalate. Similarly, this research examines the application of such ANN models to signal early indications as to whether treatments are being effective.

The precursor to Pulmonary Embolism is thrombosis, formation of blood clots inside a blood vessel obstructing the flow of blood. There are three causal elements that lead to blood clot disposition. These are known as the Virchow's triad (Virchow 1856)[5]:

- Abnormal blood flow. Abnormal blood flow is affected by narrowing of the vessels. Narrowing of the blood vessels causes turbulence that lead to formation of blood clots.
- Injuries to the vascular endothelium. Injuries to the vascular interior wall can be caused by damage to the veins arising from surgery or hypertension.
- Abnormal constitution of blood. The constituents of blood, such as proteins, water components and other elements are out of balance increasing blood thickness and propensity to form clots.

A rubric for calculating patient's risk of developing Pulmonary Embolism (PE) is known as Wells score (Wells, Anderson, et al. 1997). Wells score provides the probability that a patient might develop pulmonary embolism. It uses the following criteria:

- Are there clinical signs and symptoms of DVT?

- Is pulmonary embolism the top diagnosis?
- Is heart rate over 100?
- Is patient immobilized at least 3 days?
- Was a surgical procedure done in the last 4 weeks?
- Was patient previously diagnosed with PE or DVT?
- Is patient Hemoptysis (coughs-up blood)?
- Does patient have malignancy with treatment within 6 months or is palliative?

The goal of predicting DVT is to use data, measurements of the pre-cursors or risk factors to provide predictions about blood clot formations. The research hypothesis is that one can predict DVT in advance using data about patient's clinical data.

Finally, Deep Vein Thrombosis (DVT) is a condition that often occurs with patients with long periods of rest in hospitals. A DVT is a blood clot that forms in a vein deep in the body, often in the lower leg or thigh. A blood clot in a deep vein can break off and travel through the blood stream. The loose blood clot is called embolus. When the clot reaches the lungs and blocks blood flow, the condition is called pulmonary embolism (PE).

When PE is severe it causes lungs to collapse and leads to heart failure. One in every hundred people who develop DVT dies. According to some estimates, more than 900,000 Americans develop DVT each year and 500,000 of them develop PE with 30% of those cases being fatal. About two-thirds of all DVT events are related to hospitalization. The National Quality Forum (NQF) in its 2006 update reports that DVT is the third most common cause of hospital-related deaths in the US and the most common preventable cause of hospital death.

The data collected in this case study is typically gathered over several days. The data set includes a wide range of qualitative and quantitative clinical test results and reports.

One aim of this research was to determine viability of ANN models to predict patients' susceptibility to acquire DVT/PE.

According to StopDVT.org (2011), the risk factors for DVT/PE are the following:

- Age: over 40 years
- Already had blood clots
- Family history of blood clots
- Suffering from or had treatment for cancer
- Certain blood diseases

- Being treated for heart failure and circulation problems
- Experienced recent surgery in particular in the hips or knees
- Have inherited clotting tendency
- Who are very tall

DVT is also common among women who are:

- Pregnant
- Recently had a baby
- Taking contraceptive pill
- On hormone replacement therapy or HRT

The progress of acquiring DVT/PE is shown in Figure 29. The important factors include the type of surgery, length of surgery, whether the patient was put on chemical prophylaxis and whether mechanical (SCD) devices were employed and how well the chemical prophylaxis were administered during the patient stay.

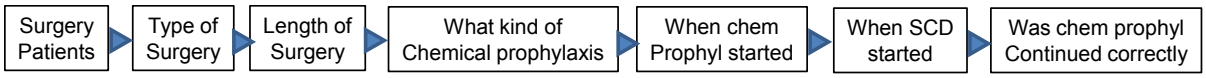


Figure 29: Progression of patients who acquire DVT/PE

The initial study considered a sample of 1,073 patients of which 225 (approximately 21%) had developed DVT/PE. Three different off-the-shelf artificial neural network tools for this study were evaluated using this data set. Of the three different commercial tools evaluated, NeuroSolutions from NeuroDimensions, Inc, was selected as the final tool of choice. The reasons for selecting this tool are explained in Appendix D, but a brief list of advantages offered by this tool include: support for wide variety of ANN algorithms, integration with Excel, ability to handle supervised learning, ability to perform K-fold cross validation, and extensive post-test accuracy and cross-validation results that it provides for the researcher after each training session.

The input data consisted of 24 different dimensions based on patient demographic and clinical elements such as: AGE, WEIGHT, GENDER, Encounter type (Outpatient, Inpatient), length of stay, Stay over 48HOURS (TRUE or FALSE), ICU vs. Acute Care patient, BMI (BioMass Index) Level, Blood measures (Platelets count, RBC, Hematocrit, Hemoglobin), other blood related values obtained from lab test results, International Normalized Ratio, an indicator of coagulation (INR of 2-3 is preferred but varies by patient),

Glucose levels, and related test results, and DVT/PE Result (1 for positive, 0 for negative). For the specific case study, a sample data from patient electronic medical record systems was collected, and then completely anonymized so there was no identifiable information.

Once the data was collected, it was loaded into an Excel spreadsheet so the four neural network models could be built using the NeuroSolutions as shown in Figure 30.

Figure 30: Starting session of Neural Solutions

To build an ANN model, a typical neural tool undergoes 4 stages for data analysis and training:

1. Data set Manager. In this stage the scope of data fields for the model are set
2. Train. The model is trained to identify its internal neural network weights
3. Test. This stage determines how robust the model is given the set of data
4. Predict. In this stage, the trained model is applied to a new patient data set for prediction.

The Excel spreadsheet plus the Neural Solutions program are shown in Figure 31. As this figure illustrates the top tool bar consists of the four functions for data selection, training, test and predict. Below the tool bar, the sample data and prediction result are shown. As illustrated in Figure 31, the model highlights (in bold face font) those input variables that are deemed significant to classification and the rest are not highlighted.

Once the model was trained on historical patient data, the trained model was applied to a new set of data for new patients during their stay in the hospital. The model has the ability to predict each patient's propensity to develop DVT/PE. The predictions are denoted by a '1' to denote patient is at risk of developing DVT/PE or '0' indicating that the DVT/PE risks are very low for the patient. When the prediction indicates risks of developing DVT/PE, then certain interventions such as medications and physical means are prescribed by the physician to the patient.

ANN models are known for their resilience to missing data. The model can fill-in the fields with missing data items. The same model can be setup to run periodically every few minutes (or every hour) for several days on the same patient but on new datasets as they're generated from the electronic medical record.

Furthermore, the model provides the input weights that it applies to compute predictions. Knowing the weights is helpful in several ways:

1) it can point to the importance of certain input variables over other variables and improves our understanding of which factors contribute to DVT/PE the most, and

2) aid in developing hypotheses for further studies that enhance evidence-based medicine, in particular studies that determine which intervention methods have been most successful in preventing DVT/PE.

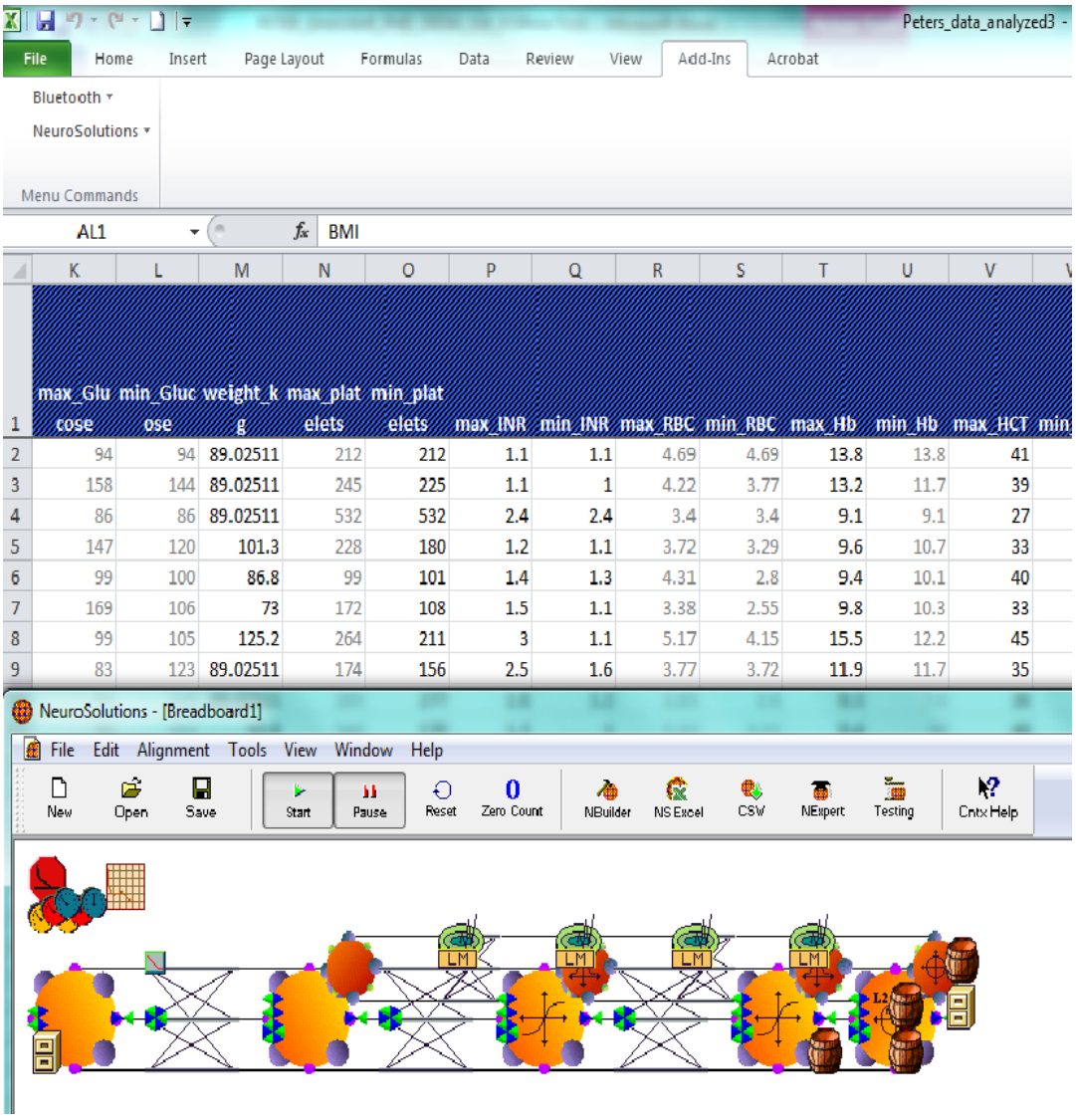


Figure 31: Example of dataset and NeuroSolutions model development environment

In the training of the neural net three factors were considered: error calculation, topology selection and prevent over-training. Error measures were computed as Mean Squared Error over all the training cases, in other words the mean squared difference between the correct answer and the answer given by the net. Through classification, the result is more than one output for each training case (one output corresponding to each dependent category). The tool allows computing the Mean Squared Error over all the outputs for all the training cases in comparison to the desired output values.

The topology is determined based on the best net configuration that produces the best training result. A typical network consists of a single hidden layer. The model automatically adds a number of neurons in each layer and additional layers to determine which topology

learns the relationship between the independent variables and the dependent variable (response) the best (by having the lowest error). By default the model uses 2- to 6 hidden layers. Larger models could take several hours to train. But, once the model is trained, predictions can be computed in a few seconds. Most models can be trained in two hidden layers.

Overtraining occurs when the number of iterations increases beyond the initial training such that the model's synaptic weights and topology match the problem specifically and the model is no longer generalizable to other datasets, namely the model does not apply to cases not included in the training. One approach to avoid "over-training" is the test-while-training method. In this approach the model is tested immediately after each iteration of training then the error gets measured. If the error starts to grow, it's indication that the researcher is starting to over-train the model.

The detailed description of input data types are presented in Table 7. As shown in the table, each data field is either independent numeric or independent category, except for one data type that represent the dependent variable. The goal of the model is to predict the dependent variable (DVT/PE Result). The third column indicates which input variables were selected by the initial training models as significant variables. These significant variables were used as input variable to train the final four models.

There are four steps for building and running an ANN model.

- Step1: Define and manage the input layer data. Define the data types, independent variables and the dependent variable.
- Step 2: Train the model on a sample of cases (typically 100 cases are sufficient for training, but more cases will help reduce percent of bad predictions)
- Step 3: Test the model using the same set of training cases plus additional cases.
- Step 4: Run the model. Observe the error rate and percent of bad predictions.

Table 7: Input Data fields and their type

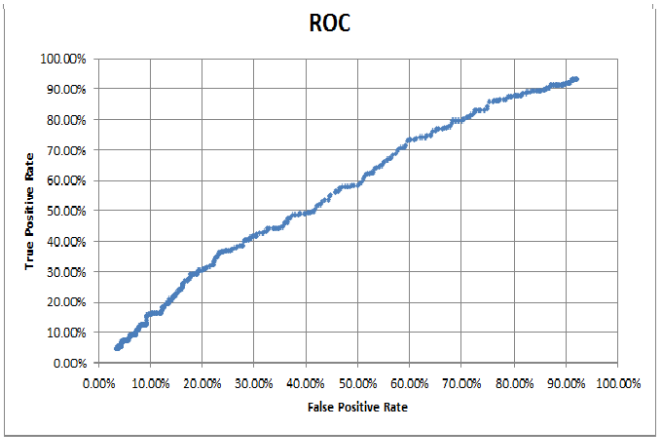
Data Variable	Variable Type	Significant to ANN Training
CASE_ID	Not Used	
CREATED_DT	Not Used	
VISIT DATE	Not Used	
AGE	Independent Numeric	Yes
Length of Stay (days)	Independent Numeric	Yes
Weight (Kg)	Independent Numeric	Yes
BMI Index	Independent Numeric	Yes
IS INPATIENT?	Independent Category	
Is Adult?	Independent Category	
Maximum Glucose	Independent Numeric	
Minimum Glucose	Independent Numeric	
Maximum Platelets	Independent Numeric	
Minimum Platelets	Independent Numeric	Yes
Maximum INR	Independent Numeric	Yes
Minimum INR	Independent Numeric	Yes
Maximum RBC	Independent Numeric	
Minimum RBC	Independent Numeric	
Maximum Hb	Independent Numeric	Yes
Minimum Hb	Independent Numeric	
Maximum HCT	Independent Numeric	Yes
Minimum HCT	Independent Numeric	
Maximum MCH	Independent Numeric	
Minimum MCH	Independent Numeric	
Maximum MCHC	Independent Numeric	
Minimum MCHC	Independent Numeric	
Maximum RDW-CV1	Independent Numeric	
Maximum RDW-CV2	Independent Numeric	Yes
Minimum Diastolic BP	Independent Numeric	Yes
Maximum Diastolic BP	Independent Numeric	
Maximum Systolic BP	Independent Numeric	
Minimum Systolic BP	Independent Numeric	
Smoker?	Independent Category	Yes
DVT/PE RESULT	Dependent Numeric	

A typical output screen of model training, cross validation and testing along with results are presented in Figure 32 as example. Note the mix of numeric and categorical independent variables. The dependent variable is the DVT/PE Result column. There were 1,073 independent patient cases in the data set, of which there were 225 confirmed positive

cases of DVT. The algorithm trained on 1,073 cases. Then it was tested for accuracy on an N-Leave-Out method, using 2% of data cases in each iteration for cross validation. The results of one model are shown in Figure 32 only as illustration of typical output. A typical output of cross validation shows a confusion matrix, ROC curve and other calculations including sensitivity and specificity for a range of thresholds. A detailed explanation and synthesis of results are covered in Chapter 5.

Threshold = 0.3		
Output / Desired	dvtPeResult(0)	dvtPeResult(1)
dvtPeResult(0)	683	157
dvtPeResult(1)	165	68

Performance	dvtPeResult(0)	dvtPeResult(1)
MSE	0.094149096	0.598103294
NMSE	#N/A	#N/A
MAE	0.223335797	0.740626924
Min Abs Error	0.001326899	0.031017633
Max Abs Error	1.055555556	1.055555556
r	#N/A	#N/A
Percent Correct	80.54245283	30.22222222



ROC Detection Threshold	Total Detections	True Positive (TP)	False Positive (FP)	True Negative (TN)	False Negative (FN)	Detected as Positive ((TP+FP)/(TP+FP +TN+FN))	False Positive Rate (FP/(FP+TN))	True Positive Rate (TP/(TP+FN))	False Discovery Rate (FP/(FP+TP))
0.001	991	210	781	67	15	92.36%	92.10%	93.33%	78.81%
0.002	991	210	781	67	15	92.36%	92.10%	93.33%	78.81%
0.003	989	210	779	69	15	92.17%	91.86%	93.33%	78.77%
0.004	988	210	778	70	15	92.08%	91.75%	93.33%	78.74%
0.005	985	210	775	73	15	91.80%	91.39%	93.33%	78.68%
0.006	985	210	775	73	15	91.80%	91.39%	93.33%	78.68%
0.007	985	210	775	73	15	91.80%	91.39%	93.33%	78.68%
0.008	985	210	775	73	15	91.80%	91.39%	93.33%	78.68%
0.009	984	209	775	73	16	91.71%	91.39%	92.89%	78.76%
0.01	984	209	775	73	16	91.71%	91.39%	92.89%	78.76%
0.011	984	209	775	73	16	91.71%	91.39%	92.89%	78.76%
0.012	983	209	774	74	16	91.61%	91.27%	92.89%	78.74%
0.013	983	209	774	74	16	91.61%	91.27%	92.89%	78.74%
0.014	983	209	774	74	16	91.61%	91.27%	92.89%	78.74%
0.015	982	209	773	75	16	91.52%	91.16%	92.89%	78.72%
0.016	980	209	771	77	16	91.33%	90.92%	92.89%	78.67%
0.017	979	208	771	77	17	91.24%	90.92%	92.44%	78.75%

Figure 32: Output screen from NeuroSolutions after training on 1,073 patient cases

Four different algorithms were used to train four independent models on the same data set. The data was randomized by the ANN tool in the first step. In the second step, a Greedy Search ANN algorithm was used to identify the significant input variables. The number of input variables was reduced from 29 to 12 according to the selection made by the first stage ANN model.

In the case of multi-layer perceptron (MLP) algorithm, the search for the optimum training led to finding the optimum number of layers. The tool was configured to find the

optimum number of layers as it constructed several multi-layer networks and compared the Mean Square Error of the MLP networks. The model configurations consisted of 2-node, 3-node, 4-node, 5-node and 6-node arrangements. The tool selected the optimum number of layers that provided the lowest MSE. The largest number of iterations occurred with the 6-node model with 150 iterations. The model completed training and running in 40minutes on a dual core Intel processor desktop computer (2.8GHz CPU speed).

One of the advantages of NeuroSolutions is that it presents an object oriented graphical representation of the ANN model being built. The model representation for Probabilistic Neural Network (PNN), Support Vector Machine (SVM), Multi-Layer Perceptron (MLP) and General Feed-forward network (GFN) are shown in Figures 33-36 respectively.

PNN Model Description: As shown on the model breadboard at the bottom of Figure 33, the PNN model consists of several computational objects. From left to right, they are: the input axon, followed by a summation of weights multiplied by input, then the Gaussian calculator followed by the output weights followed by the output layer. The final object is the error criteria analyzer that compares the computed result against desired output.

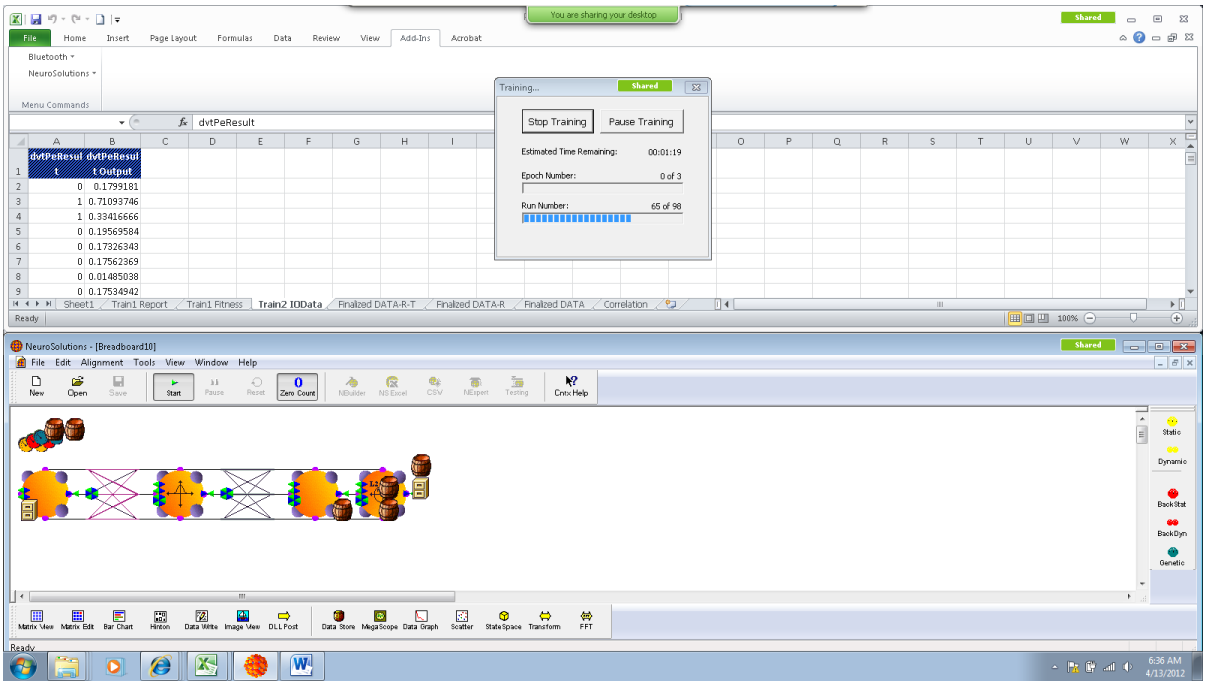


Figure 33: PNN Network model illustration during training and cross-validation

SVM Model Description: The SVM model consists of several computational objects as shown in Figure 34. From left to right along the bottom, they are: the input axon, followed by a summation of weights multiplied by input, then the Gaussian calculator followed by the output weights followed by the output layer. The final object is the error criteria analyzer that compares the computed result against desired output. The red back-propagation objects are used to calculate the error sensitivities for each of the network weights of the output layer. The green icon with the SVM implements the adatron learning algorithm used by SVMs. This updates the network weights using the error sensitivity information contained in the back-propagation objects.

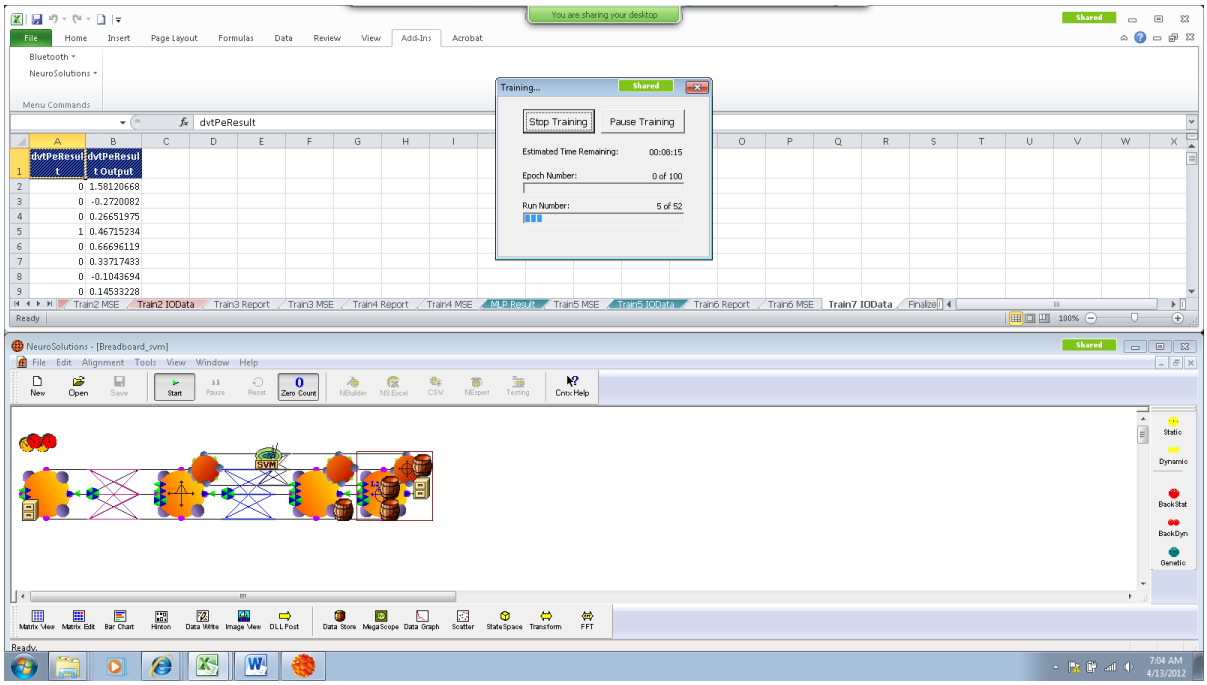


Figure 34. SVM model illustration during training & cross-validation

MLP Model Description: As shown on the bottom of Figure 35, the MLP breadboard model consists of several computational objects. From left to right along the bottom, they are: the input axon, followed by a summation of weights multiplied by input, then the hidden hyperbolic tangent layer followed by a set of hidden layer weights, followed by the hyperbolic tangent output layer. The final object is the error criteria analyzer that compares the computed result against desired output. The red back-propagation objects are used to calculate the error sensitivities for each of the network weights. The green icons with the LM

implement the Levenberg-Marquardt algorithm. This updates the network weights using the error sensitivity information contained in the back-propagation objects.

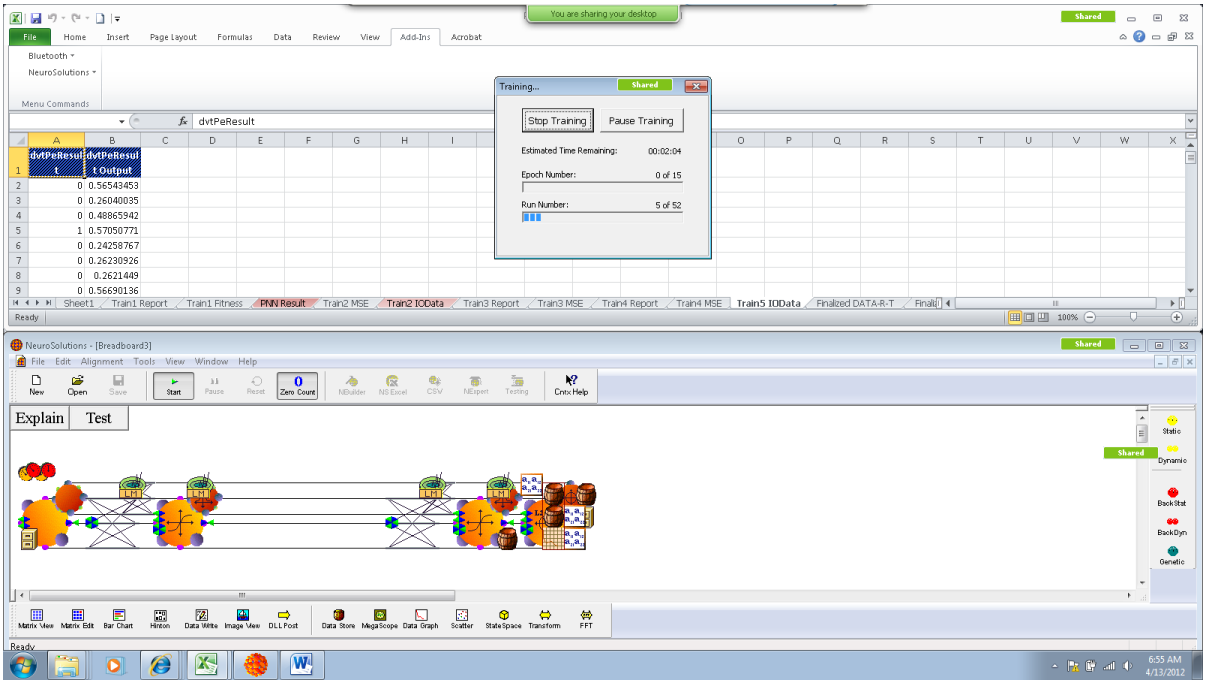


Figure 35. MLP model illustration during training & cross-validation

GFN Model Description: The General Feed-forward Network (GFN) model consists of several computational objects as shown in Figure 36. From left to right along the bottom, they are: the input axon, followed by a summation of weights multiplied by input, then the hidden hyperbolic tangent layer followed by a set of hidden layer weights, followed by the hyperbolic tangent output layer. There is also a bypass layer of weights that goes directly from the input layer to the output layer. This layer finds the linear relationships in the data. The final object is the error criteria analyzer that compares the computed result against desired output. The red back-propagation objects are used to calculate the error sensitivities for each of the network weights. The green icons with the LM implement the Levenberg-Marquardt algorithm. This updates the network weights using the error sensitivity information contained in the back-propagation objects.

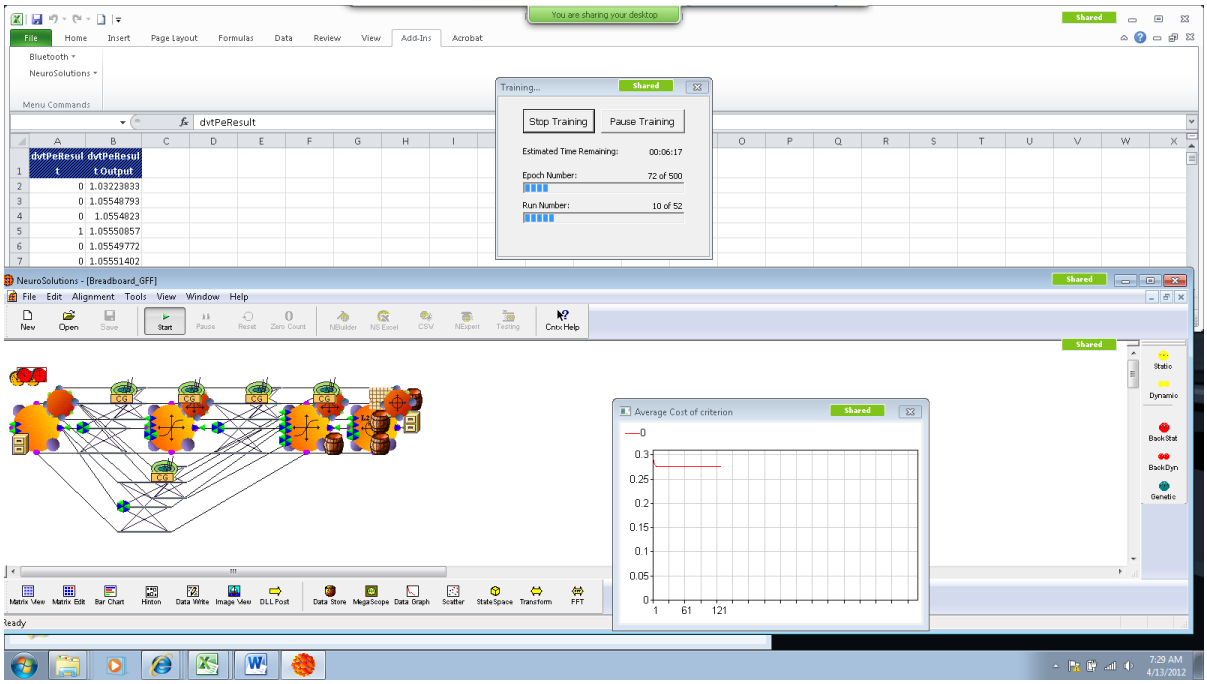


Figure 36. General Feed-forward network during training & cross-validation

Training each model required separate training, cross-validation and testing process. The models consumed several "epochs" to complete the training. An epoch is representation of an entire training set in neural networks, namely the number of iterations of training required for the model to reach its global optimum solution. The model training stops when the change in Mean Square Error (MSE) reaches a small threshold defined by the user. Table 8 shows the number of epochs that each model took for training. This table compares the relative efficiency of each neural network model.

Table 8. Computational Resources consumed by each model

Model	Epochs
MLP- LM	20
GFN-LM	20
SVM	150
PNN	3

A sensitivity analysis was performed to determine the significance of input variables. This testing process provides a measure of the relative importance among the inputs of the neural model and illustrates how the model output varies in response to variation of an input.

Sensitivity analysis works by taking an input and varying it between its mean, ± a (user-defined) number of standard deviations while all other inputs are fixed at their respective means. The network output is computed for a user-defined number of steps above and below the mean. This process is repeated for each input and the impact on output is recorded at each step. An alternate variation of this process is to vary the input of interest between its minimum value and its maximum value. This option is especially useful for binary inputs or inputs which have a non-Gaussian distribution.

The result of sensitivity analysis is shown in Figure 37. This figure shows the relative strength of weights of input variables. Of these variables, Minimum and Maximum INR, followed by patient Weight, Length of stay, Maximum RDW-CV2, Patient age were most significant input variables toward classification of patients.

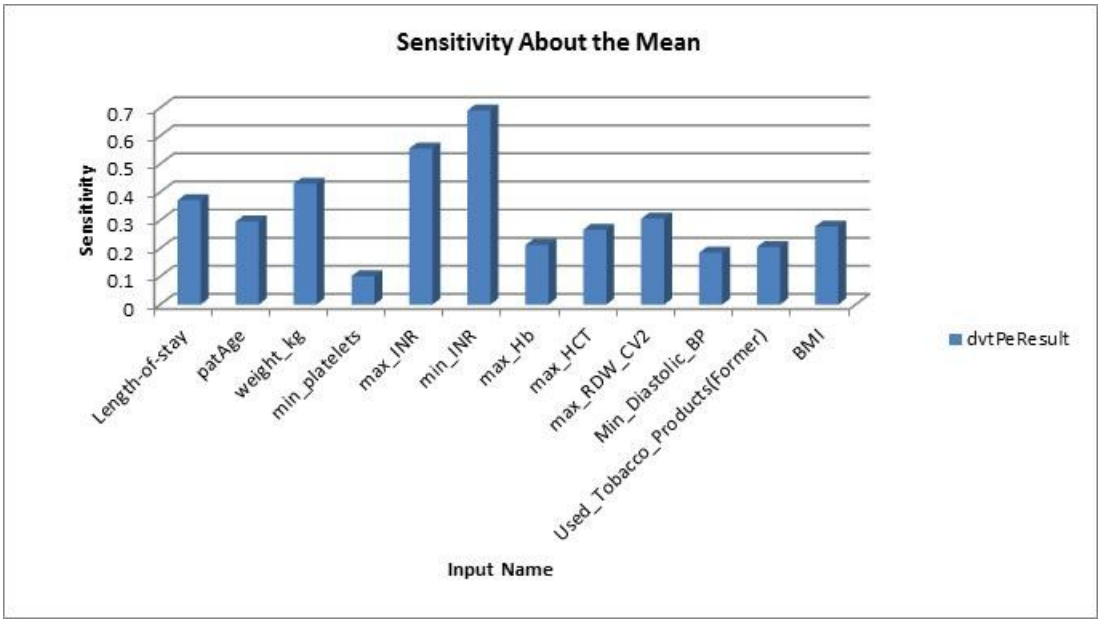


Figure 37: Strength of Input variables towards classification

The results point to several variables as being significant in predicting DVT/PE. The sensitivity analysis provided the following ranking of the input variables based on their significance towards patient classification:

1. Minimum INR
2. Maximum INR
3. Weight
4. Length of Stay

5. Maximum RDW-CV2
6. Patient Age

Sensitivity analysis performed using NeuroSolutions software package only reveals the degree that one variable impacts the output. It does not determine if the changes in input are positively or negatively correlated. A clinical interpretation of the sensitivity analysis can be given as follows: The INR values explain the patient's blood characteristics. It makes sense that sensitivity analysis has pinpointed Minimum and Maximum INR as significant inputs to classification. INR, a standard measure of blood clotting characteristics indicates the blood tendency to form clots. A lower INR correlates with higher chance of blood clot formation. The significance of RDW-CV2 can be explained as it's a property of red blood cells that correspond to the width of red blood cells. Wider blood cells are more likely to be caught in capillaries, blocking blood flow. Length of Stay is significant as it's empirically observed that longer patient's stay in hospital is correlated with increase in frequency of DVT, but only for a certain length of stay. The relationship between Length of Stay and occurrence of DVT follows a quadratic equation. Patient Weight might point to certain underlying patient characteristics which could related to the patient's lower levels of mobility or other factors yet unknown. Of these variables, the blood related measurements point to possible opportunities for clinical intervention through medication.

But, the benefit of sensitivity analysis is that it highlights those significant variables which can be further studied in future research to determine causal relevance to a particular disease. Unlike prior research such as Well's CPR method that choose input variables in somewhat arbitrary method, neural network models constructed in this research can reveal a data-driven approach through sensitivity analysis to identify the significant input variables among a large list of input variables.

Future areas of research could include clinical validation of results and provide clinical interpretations of the input variables. These areas are discussed in more detail in Chapter 6.

Chapter 5

5 ANALYSIS OF RESULTS

The clinical case study consisted of 1,073 patient cases who were admitted to a hospital for various treatments of which 225 had developed positive cases of DVT. The patient data consisted of 29 independent variables and one dependent variable. The input data included various relevant physical and vital sign data ranging from blood pressure to heart rate and blood lab test results. The input variables consisted of both continuous and dichotomous variables. The dependent variable was a dichotomous variable that represented the clinical outcome, the occurrence or absence of a disease. In this study, the output was defined by a marker called Deep Vein Thrombosis (DVT).

DVT is the formation of blood clots in deep veins, typically in leg veins. Blood clots can dislodge and flow to lungs causing a more critical condition called Pulmonary Embolism (PE). DVT/PE is a serious medical condition that can cause serious pain and even death. In the US alone approximately 350,000 to 600,000 patients suffer from DVT and at least 100,000 deaths per year are attributed to DVT/PE (The Surgeon General's Call to Action to Prevent Deep Vein Thrombosis and Pulmonary Embolism, 2008).

Neural networks have been successfully applied to classify patterns based on learning from prior examples. Different neural network models use different learning rules, but in general they determine pattern statistics from a set of training examples and then classify new data according to the trained rules. Stated differently, a trained neural network model classifies (or maps) a set of input data to a specific disease from a set of diseases.

Four models were trained and tested in two stages: in the first stage, a genetic, greedy search neural network algorithm was employed to identify the input variables with most predictive power. It narrowed the list of input variables from 29 down to 12 variables. In the second stage, all four models were trained and tested on the 12 input variables that were selected by the genetic algorithm model (obtained from stage 1). The list of the most predictive variables was provided in Section 4.4 via Table 7. Next in Table 9, a description of significant input variables is provided.

Table 9: Input variables description

Input Variable	Data Type	Definition
AGE	Continuous	Patient's age
INPATIENT	Dichotomous	Is patient admitted as inpatient?
WEIGHT	Continuous	Weight during stay in Kg.
MIN PLATELET	Continuous	Minimum no. of blood platelets, tiny cells that assist in blood clotting
MIN INR	Continuous	Minimum INR (International Normalized Ratio). The standard for a healthy person is 1.
MAX INR	Continuous	Maximum INR (International Normalized Ratio).
MAX Hemoglobin	Continuous	Maximum Hemoglobin concentration. The average for humans is 16 g/100ml.
MIN DIASTOLIC BP	Continuous	Minimum blood pressure when heart is at rest. A normal diastolic BP is under 80, but over 90 is considered hypertension.
MAX HCT	Continuous	Maximum hematocrit: the proportion, by volume, of red blood cells
MAX RDW CV2	Continuous	Minimum red blood cell distribution width.
BMI Index	Continuous	BioMass Index, a measure of weight and height, values between 18.5-24.9 are regarded normal weight
SMOKER	Categorical	Patient's smoking status as either Former, Unknown, or Current Smoker

5.1 Computational Method

In this research, four different prediction and classification algorithms were trained and tested on 12 data input variables and 1,073 patient cases. There were 89 true positive cases in the retrospective study. This research used NeuroSolutions version 6.0 (NeuroDimension, Inc. 2011) neural network software package to build and test the models. A detailed description of this software package appears in Appendix D.

For each of the four models, the "Leave-N-out" technique was employed. This technique is a combined training and cross-validation method used to minimize bias due to random data selection. This approach trains the network multiple times, each time omitting a different subset of the data and using that subset for validation. The outputs from each tested subset are combined into one testing report and the model is trained one final time using all of the data.

The test results of all four models can be compared using classification measures such as number of false positives (FP), false negatives (FN), true positives (TP) and true negatives (TN). The performance of four ANN models is shown in Table 10.

Table 10: Model test results

Model	TP	FP	TN	FN	Total
Probabilistic Neural Network	94	216	632	131	1,073
Support Vector Machines	98	230	618	127	1,073
Multi-layer Perceptron with LM	90	213	635	135	1,073
Generalized Feed forward with LM	129	336	512	96	1,073

5.2 Accuracy and Validation

External validity of medical prediction models is an extremely challenging task. Clinical validation is challenging not just because it involves prospective patient studies, double-blind studies and careful administration of research protocols, but for two other reasons: first, if the model predicts a disease and the patient gets the treatment per recommendation of the predictive model, we can't determine if the patient would have exhibited the predicted disease to confirm our prediction. In other words, the medical treatment masks the possible outcome. Second and in contrary, if the model predicts no disease but the patient gets treatment, we would not be able to invalidate the model's prediction since we can't claim that the disease might have occurred.

This research focuses on internal validity in terms of accuracy but leaves external (clinical) validation to future research projects. Several measurements have been proposed as methods for internal validation. Some of the measurements that are commonly used to compare accuracy of classification models include: Accuracy, Sensitivity, Specificity, Area Under Receiver Operating Curve (AUROC) and Likelihood Ratio (LR). Sensitivity measures the fraction of positive cases that are classified correctly as positive. Specificity is the fraction of negative cases that are classified correctly as negative. AUROC is the area

under the ROC and is regarded as a good overall measure of predictive accuracy of a model (Bewick, Cheek, Ball 2004). An ROC can be plotted by connecting the points obtained from ANN model results at different model thresholds as shown in Figure 38. A ROC is a graph that represents a plot of sensitivity versus (1 - specificity). The Area under ROC curve (AUROC) can be computed by the sum of trapeziums areas under the curve . An AUROC close to 1.0 is a considered an excellent discrimination, but a value near 0.50 suggests no discrimination (similar to a coin flip).

The ROC curve for each model was computed and compared as shown in Figure 38. From a visual inspection, it's clear to see that the SVM model has a more desirable accuracy due to its larger relative area under the ROC (AUROC).

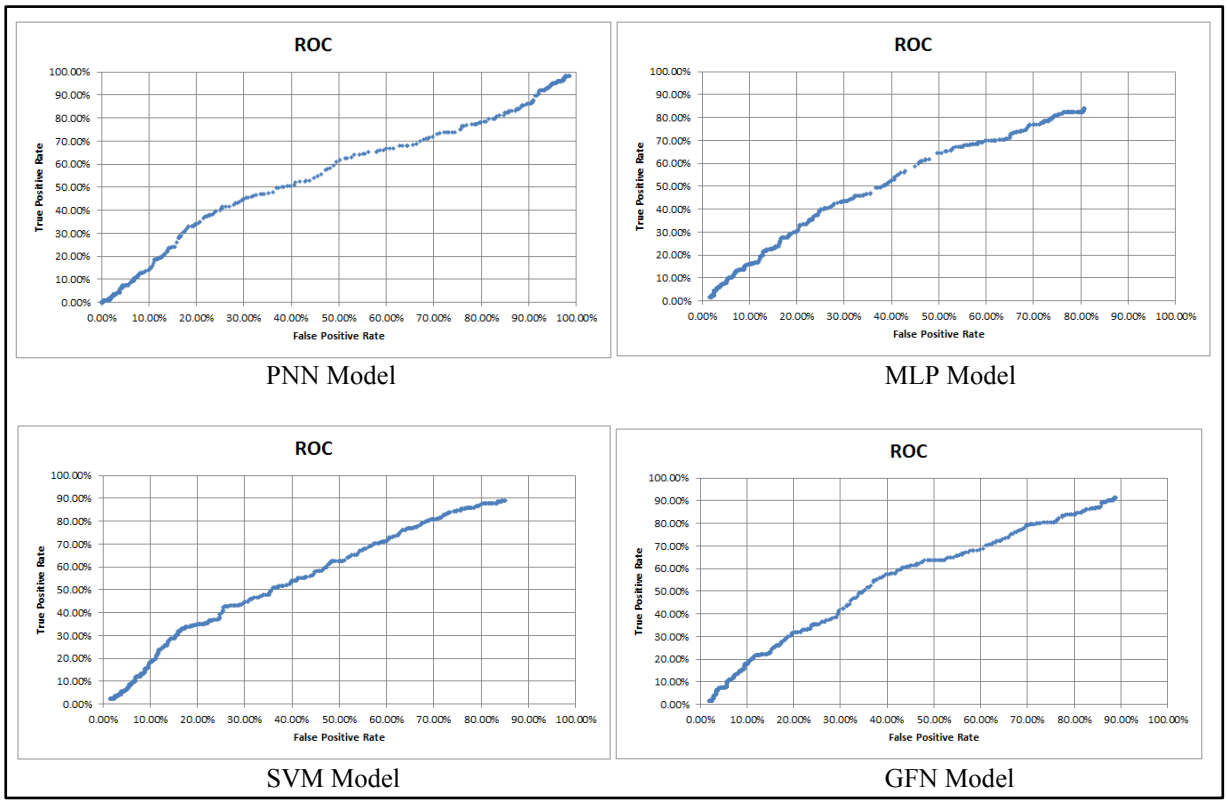


Figure 38. ROC Curves for the four ANN models

Likelihood ratio combines both sensitivity and specificity into a single measure. It provides the direct estimate of how much a test result will change the odds of having a disease. The Positive LR (LR+) shows how much the odds of the disease increase when a test is positive. The Negative LR (LR-) shows how much the odds of the disease decrease

when a test is negative. Odds can be derived from probability. To convert from probability to odds, divide the probability by one minus that probability.

Positive LR is the ratio of sensitivity to one minus specificity (Delen 2009). The accuracy measures may be defined as:

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{89}$$

$$sensitivity = \frac{TP}{TP+FN} \tag{90}$$

$$specificity = \frac{TN}{TN+FP} \tag{91}$$

$$LR+ = \frac{sensitivity}{1-specificity} = \frac{\Pr(T+|D+)}{\Pr(T+|D-)} \tag{92}$$

$$LR- = \frac{1-sensitivity}{specificity} = \frac{\Pr(T-|D+)}{\Pr(T-|D-)} \tag{93}$$

An LR+ is a ratio, equal to the probability of a person who has the disease and tested positive divided by the probability of a person who does not have the disease and tested positive. An LR- is another ratio, equal to the probability of a person who has the disease and tested negative divided by the probability of a person who does not have the disease and tested negative.

Likelihood ratio is useful when the pre-test odds of having a disease are known. Then, the post-test odds of disease can be computed by:

$$odds_{post-test} = odds_{pretest} * likelihood\ ratio \tag{94}$$

This calculation is based on Bayes Theorem. One can convert odds to probability simply by (95).

$$Probability_{pretest} = \frac{(TP+FN)}{Total\ Sample}$$

Alternatively,

$$Odds_{pretest} = \frac{(Probability_{pretest})}{(1-Probability_{pretest})} \tag{95}$$

When a model uses continuous data measurements, then different thresholds may be applied in order to decide which value is the cut-off to distinguish between patients with disease. The best model has the highest values for sensitivity and specificity. In certain situations, both may not be equally important. For example, a false-negative (FN) prediction might be more critical than a false-positive (FP) prediction. If no preference is given to either measurement then, Youden's index (*J*) may be used to choose an appropriate cut-off, computed by (Bewick, Cheek, Ball 2004). When using ANN models to make a binary prediction the result is a continuous measure that varies from 0.00 to 1.00. The ideal threshold that would classify patients into disease or healthy can be set using the Youden's index *J*. At the point where Youden's index is highest, the threshold can be set at that level. The Youden's index for each of the four ANN models were computed and used to determine the ideal threshold. The relationship between Youden's index *J* and sensitivity and specificity is defined as:

$$J = sensitivity + specificity - 1 \tag{95}$$

Higher value of *J* is desired. The maximum value that *J* can take is 1, when the test is perfect.

PPV corresponds to the number of true positives divided by the sum of true positives and false positives. NPV is computed as the ratio of

$$PPV = \frac{True\ Positives}{(True\ Positives + False\ Positives)} \tag{96}$$

$$NPV = \frac{True\ Negatives}{(True\ Negatives + False\ Negatives)} \tag{97}$$

Figure 39 shows a statistical truth table (Also known as Confusion Matrix) that illustrates how sensitivity, specificity, PPV and NPV are related. In this figure, results of the GFN model are used only as illustration to show how PPV, NPV, Sensitivity and Specificity are calculated.

		Truth (Condition) as determined by "Gold Standard"		
		Positive Condition	**Negative Condition**	
Test Outcome	Test Outcome: **Positive**	**True Positive** (TP) = 129 1 - β	**False Positive** (FP) = 336 (Type I error) A	Positive Predictive value = $\frac{\sum True\ Positive}{\sum Test\ Outcome\ Positive}$ = TP/(TP+FP) =129/(129+336) **=27.7%**
	Test Outcome: **Negative**	**False Negative** (FN) = 96 (Type II error) β	**True Negative** (TN) = 512 1 – α	Negative Predictive value = $\frac{\sum True\ Negative}{\sum Test\ Outcome\ Negative}$ =TN/(FN+TN) =512/(96+512) **=84.2%**
		Sensitivity $\frac{\sum True\ Positive}{\sum Condition\ Positive}$ =TP/(TP+FN) =129/(129+96) **=57.3%**	**Specificity** $\frac{\sum True\ Negative}{\sum Condition\ Negative}$ =TN/(FP+TN) =512/(336+512) **=60.4%**	

Figure 39. Statistics Truth Table (Confusion Matrix)

In statistics and Medicine, Gold Standard Test refers to a diagnostic test that is best available to diagnose or classify a patient into either disease or normal condition. Gold standard test is not necessary a perfect test, but one that offers the most accurate test possible without restrictions. The ideal Gold standard test offers 100% sensitivity and specificity. In practice, however, a Gold Standard test is less accurate. For example, to diagnose brain tumor, one can perform a biopsy or an MRI. A biopsy test is regarded as the Gold Standard for diagnosing brain tumor, but since MRI test is less accurate but a practical substitute, it's regarded as an "imperfect gold standard" or "Alloyed gold standard" (Spiegelman,

Schneeweiss, McDermott, 1996). Gold standard tests vary over time for each disease as the state-of-the art methods of diagnostic tests improve over time.

5.3 Comparison of results

All four models were optimized for classification of cases into a dichotomous dependent variable: the presence or absence of DVT. The results showed that the SVM algorithm was most accurate followed by the MLP model and the General feed-forward neural network model. All four methods are compared using the accuracy measurements in Table 11.

Table 11: Accuracy Measures of Neural network models

Measurement	Probabilistic Neural Network	Support Vector Machine	Multi-Layer Perceptron-LM	Generalized Feed-forward-LM
Accuracy	0.6766	0.6673	0.6757	0.5974
Sensitivity	0.4178	0.4356	0.4000	0.5733
Specificity	0.7453	0.7288	0.7488	0.6038
LR+	1.6402	1.6059	1.5925	1.4470
LR-	0.7812	0.7745	0.8013	0.7067
Youden's J	0.1631	0.1643	0.1488	0.1771
PPV	0.3032	0.2988	0.2970	0.2777
NPV	0.8283	0.8295	0.8247	0.8421
AUC	0.5593	0.5945	0.5760	0.4176

All four models exhibited low sensitivity measures indicating their poor ability to detect true positives. This is due to the lower number of positive DVT cases in this study (only 225 out of 1,073 cases had positive DVT cases). AUC is regarded as a better measure of performance than accuracy (Ling, Huang, Zhang 2003). Since the AUC value of SVM algorithm is highest among all four ANN algorithms, one can declare that in this instance only, the SVM algorithm has highest accuracy on this data set.

5.4 Oracle Description

Since their introduction in 1960's, various neural network algorithms have been proposed and successfully implemented to classify and predict future state of output variables. Certain models are more suitable to specific class of problems based on the type

and number inputs and output classifications. Typically, no single neural network model is best for all types of problem.

An approach that uses an ensemble of prognostic algorithms is shown to be effective in providing more accurate prediction (Hu, Youn & Wang 2010).

In this research five different ensemble methods were employed for the oracle to select from. The first ensemble used conditional logic to maximize the number of TP and minimize the number of FP predictions. Studies have shown that the Area Under Characteristic (AUC) is a better measurement of an algorithm's performance than comparing Accuracy measure (Ling, Huang, Zhang 2003). The oracle program compares the AUC values for all ensembles and ANN algorithms and selects the model or ensemble with the highest AUC value. The oracle program was written in R, an open source statistical program (Wang 2012). The program was tested to evaluate and compute AUC for all ensembles.

The results of the ensembles are shown in Table 12. A comparison of accuracy for these ensembles appears in Table 13.

Table 12: Results of the five ensemble programs

Ensemble	Schema	TP	FP	TN	FN	Total
Ensemble#1	Uses Relative preference for the more accurate model	152	426	422	73	1073
Ensemble#2	Uses TP ratio to increase TP count	105	246	602	120	1073
Ensemble#3	Reduces FN by a ratio of each model's FN count	131	345	503	94	1073
Ensemble#4	Diversity based to increase robustness	152	428	420	73	1073
Ensemble#5	Optimization based to reduce overall error	139	387	461	86	1073

Ensemble#1 took a preference ratio of four models to produce a more accurate prediction with emphasis to reduce FN count. The second ensemble combined weighted sum of predictions from each model in the ensemble. The weights were determined to maximize the number of TP predictions. Ensemble#5 selected weights by using the optimization method. The optimization model was computed using Excel Solver with one minimum

objective function and five constraints a shown in Figure 40. The optimum weights were 0.0, 0.03652, 0.5234, 0.437368 computed for PNN, SVM, MLP and GFN models respectively.

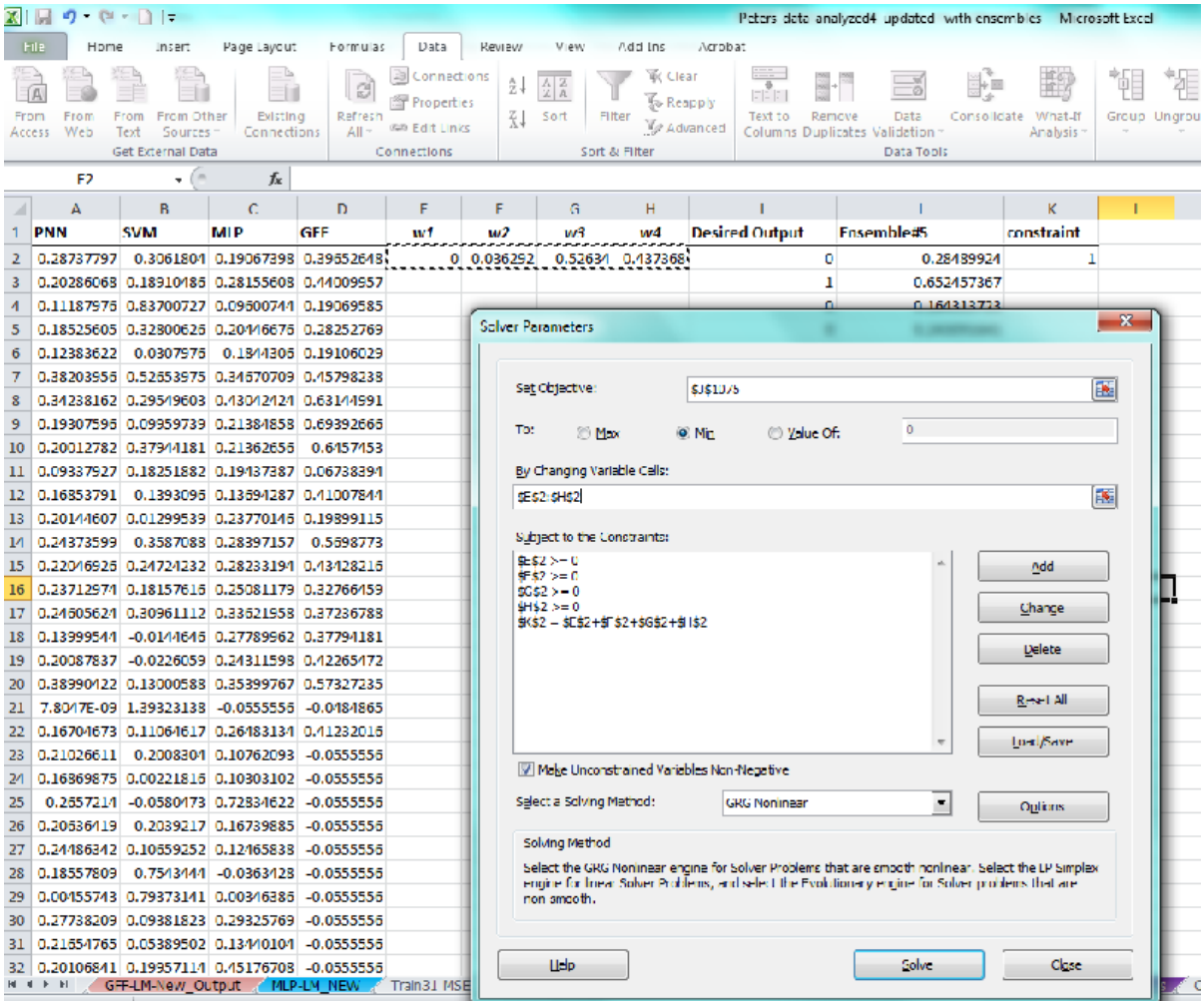


Figure 40. Excel Solver with Objective Function and Constraints for Ensemble#5

A comparison of accuracy measures among the five ensembles are shown in Table 13. One method to compare all four models and the five ensemble programs is to use the Receiver Operating Curve (ROC) plot. The ROC curve is a plot of sensitivity versus (1 – specificity), and generally is considered a good accuracy measure of binary classifiers (Bourdes, Ferrieres, Amar, Amelineau, et al, 2011).

Table 13: Comparison of Five Ensemble accuracy

Measurement	Ensemble #1	Ensemble #2	Ensemble #3	Ensemble #4	Ensemble #5
Accuracy	0.5350	0.6589	0.5909	0.5331	0.5592
Sensitivity	0.6756	0.4667	0.5822	0.6756	0.6178
Specificity	0.4976	0.7099	0.5932	0.4953	0.5436
LR+	1.3448	1.6087	1.4311	1.3385	1.3537
LR -	0.6520	0.7513	0.7043	0.6551	0.7031
Youden's J	0.1732	0.1766	0.1754	0.1708	0.1614
PPV	0.2630	0.2991	0.2752	0.2621	0.2621
NPV	0.8525	0.8338	0.8425	0.8540	0.8428
AUC	0.6035	0.6047	0.6046	0.6040	0.5943

Figure 41 shows a bar chart graph of AUC value for all models. The best prediction method would result in the higher AUC value.

The diagram illustrates two observations: The prediction results are not as accurate as one would like. This is attributed to the fact there were too few positive cases in the entire population to help train a more accurate predictive model. Furthermore, several of input variables were highly correlated such that the predictive contribution of some variables was less significant for making a more accurate prediction. However, prediction accuracy was improved using ensemble of models. In this particular data set, Ensemble #2 provides the highest level of AUC. Therefore, the oracle program would select Ensemble #2, for predicting DVT for patients in this situation. However, as new data arrives and the models get retrained or for other diseases and data sets, it's completely plausible that other Ensembles would perform better than Ensemble#2. The point of this investigation is to emphasize that the ensemble of algorithms produces better accuracy and more robustness to predict from various data sets and for different diseases.

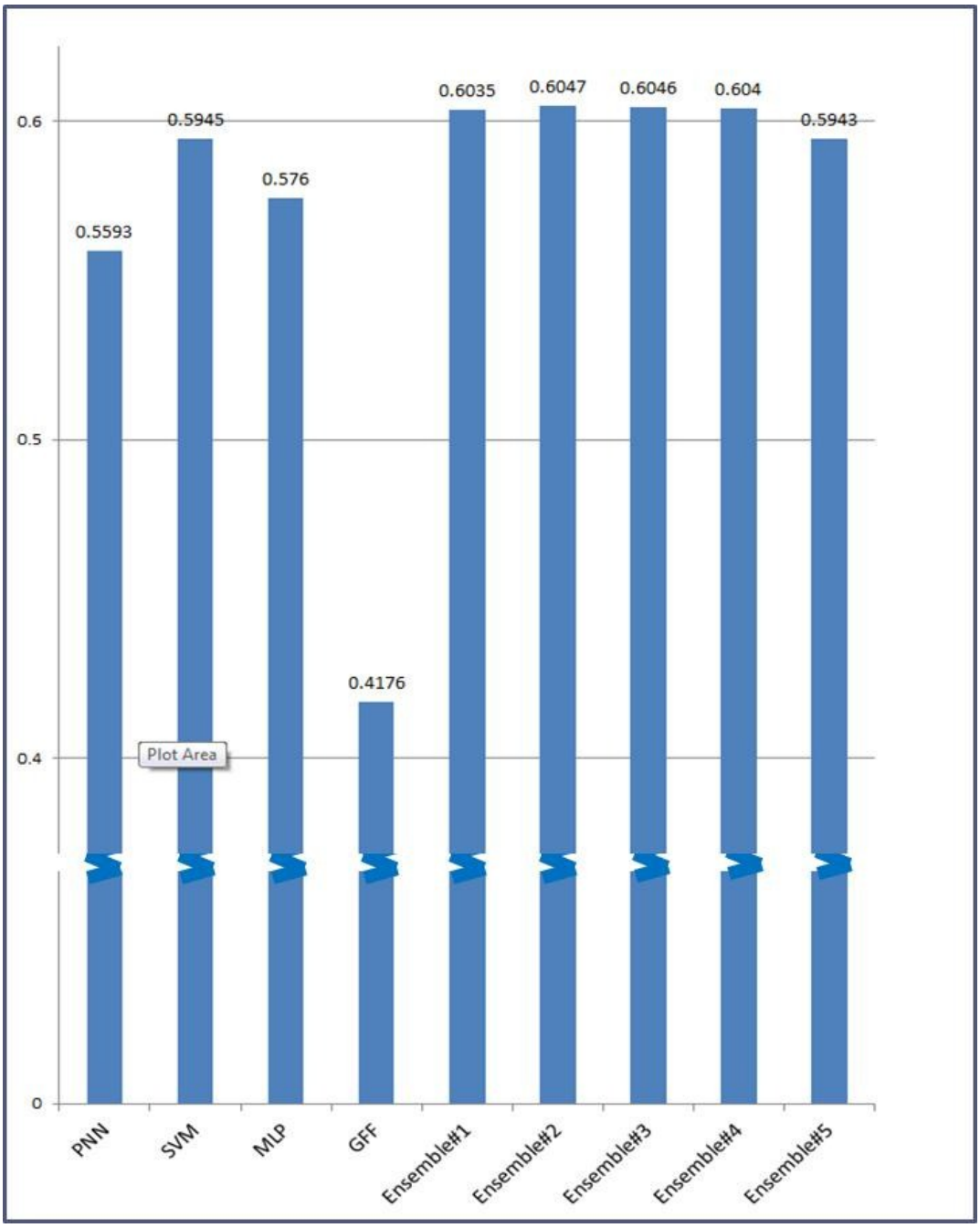


Figure 41: Bar chart graph of AUC calculation for all models and ensembles

The oracle program was written in R, an open source statistical program (Wang 2012). The source code for the oracle is shown in Appendix E. R is a statistical language for analytics. It can run both in interactive and script file environments.

This finding suggests that ensembles that optimize weights of combined models can drastically improve accuracy as each model contributes its best characteristics towards predicting the outcome.

A review of the results shown in tables 11 and 13, reveal that all ensemble models except for Ensemble#5 were more accurate than each neural network model alone, based on criteria established by the ensembles. The ensemble approach as demonstrated in this case study provides various schema that can improve robustness and accuracy of multiple neural networks. The results obtained in this research illustrate the notion that multi-model frameworks can be enhanced by using various ensemble (or committee network) constructs.

Chapter 6

6 CONCLUSION AND FUTURE RESEARCH TOPICS

6.1 Conclusion

In this dissertation, the viability and feasibility of a prognostics and health management framework using a multi-model ANN oracle to predict medical disease was examined. The Committee approved work on three research contributions that included: 1) developing a control theory model to prognostics, 2) Develop a multi-model ANN framework for disease prediction, and 3) perform feasibility analysis of the framework. The Advisory Committee agreed to these research deliverables and these deliverables have been completed.

A control theoretic framework for multi-model Artificial Neural Network was developed and presented in Chapter 2. The framework presented a feed-forward and feedback model to prediction. It presented a prognostics engine for making predictions. The advantages of feed-forward predictive models were presented in a control theoretic approach.

This research completed the requirements for developing the prognostics engine by employing four ANN models and an oracle program to select among several ensembles of ANN models. The mathematics foundation of ANN models used in prognostics engine were explained and defined in Chapters 3 and 4.

In Chapter 5, it was demonstrated how various neural network methods can be used to make independent predictions using an identical input data set. Accuracy and viability of these models were evaluated using several metrics including accuracy measures. As planned, this research evaluated performance of four ANN models and demonstrated viability of using a multi-model framework applied to a specific case study, the DVT/PE dataset of 1,100 patients.

Several rubrics for comparing accuracy model and building ensembles to enhance accuracy were developed and explained in Chapter 4. It was demonstrated that the performance of these models varies depending on the type and volume of input and output variables. Since no single model is a perfect fit for all types of prediction, it was recognized that each of the four models had certain strengths and weaknesses. It was demonstrated that

by combining multiple models one can improve classification accuracy. As was approved by the Committee, an oracle program was constructed to select the best weighted combination of results from multiple neural network models in order to enhance prediction accuracy.

This research explored and confirmed viability of the framework in 3 ways: 1) It developed a multi-model ANN prognostics engine, 2) It developed a mechanism for comparing accuracy of multiple ANN models, and 3) It devised an oracle program to select the most accurate ensemble. The viability of the framework was evaluated and tested using a realistic retrospective dataset of 1,073 patients. Four ANN models were developed and their predictions were compared with the actual patient disease condition. Five ensembles were formed to further demonstrate that accuracy can improve by combining results of multiple ANN models.

This research proposed and demonstrated that predictive frameworks based on multiple ANN models are viable and practical methods to patient disease prediction. It was further demonstrated through a case study that an oracle program or an overseer can select a more accurate prediction from among multiple models or multiple ensemble of models. Five different ensemble schemes were developed in Chapter 5. The results of the oracle program or DVT/PE data were presented in Chapter 5.

6.2 Future Research

Given the results of this research, there are several areas where the framework described in this study can be applied to enhance patient health prognostics research. These areas include:

1. Treatment Efficacy: Consider models that can also predict the effectiveness of medical treatments as an early indicator to whether the treatment is "working". One can evaluate the effectiveness of the multi-model ANN framework presented in this research to predict the efficacy of treatment given the change of patient's data over time.
2. Reduce false alarms: Studies have shown that as much as 60% of alarms in patient monitors are false and ignored by care providers. The feasibility of using ANN models to reduce false alarms would be valuable to reducing healthcare cost and improving patient care.

3. Anesthesia Reactions: Often patients develop post-anesthesia reactions such as nausea. These reactions can be prevented by medication, but not all patients are candidates for this treatment. The goal of this study would be to predict which patients are likely to develop nausea and other reactions so these reactions can be prevented by proper medication before anesthesia.
4. Risk Stratification: Using proper ANN model, one can determine the risk level associated with a patient in advance to determine the most suitable care protocol for that patient.
5. Perform in-clinical studies by providing a ready-to-use prediction tool based on the framework demonstrated in this research to physicians in order to compare the results of the predictive models against the real life data.
6. One can study to determine causal relationships and direction of causality between the significant input variables and response variable. Finally, future research can aim to show that medical interventions and decisions based on this tool have resulted in better patient care and outcomes.
7. As a future research project, it would be a worthwhile study to compare the predictions of this approach (using neural network models) against other prediction methods such as Well's scoring rules (Wells 2006) that are founded on prior medical evidence. To validate this model, other statistical tools were employed such as Chi-square and Correlation analysis. The purpose of using these statistical tools was to determine whether the independent variables were truly independent and whether there are correlations (either positive or negative) relations between the input variables.

APPENDICES – SUPPLEMENTAL CONTENT

Appendix A: Prognostics Methods

Prognostics Models:

Prognostics models can be classified into three general types (Eklund 2009; Hines 2009; Peysson et al. 2009). Type I is reliability based. It applies the traditional time to failure analysis by tracking a population of failures and using statistical methods for the estimation of reliability. Some typical life distributions that are used in this type of prognostics include Weibull, exponential and normal distributions. Type I prognostic methods does not incorporate the real time monitoring of operating conditions or environmental conditions. For example, a system that has operated under harsher environments is likely to fail faster than the system based on past environmental conditions or the past data.

The Weibull model is frequently used in type I methods because it offers flexible distributions for a variety of failure rate profiles. The two parameter Weibull model uses a shape parameter $\boldsymbol{\beta}$, and a characteristic life parameter $\boldsymbol{\theta}$. The result at time $\boldsymbol{t}$ is:

$$\lambda(t) = \frac{\beta}{\theta}\left(\frac{t}{\theta}\right)^{\beta-1} \tag{98}$$

Type I prognostic methods does not incorporate the real time, operating conditions or environment. For example, consider the life expectancy of a computer disk drive. The disk drive is known to have a failure distribution of 20,000 hours and standard deviation of 5,000 hours. A disadvantage of type I approach is that it does not consider the operating condition of the system. In addition, type I prognostics offers an average for failure rate but specific failure predictions are preferable. For example, a system that has operated under harsher environments is likely to fail faster than the mean time to failure (MTTF) for that system.

Type II methods are also known as the stressor-based approaches that consider the operational and environmental condition data. Type II methods can be used if the condition data are measurable and correlated to the system degradation. This approach includes methods such shock models and use traditional Markov methods. While this type of analysis is superior to Type I methods, it still lacks the unit-to-unit variance. This type considers the failures of a system in its operating environment to provide an average remaining life of a

component. Some of the environmental data might include temperature, vibration, humidity and load. As an example, the proportional hazard model is a type II prognostic model. Knowing the causes, one can predict reliability of a system. The simplest model in this approach is the regression model: given the operating and environmental conditions, one can predict the system failure and remaining useful life by a regression equation:

$$Failure\ rate = \beta_0 + \beta_1 \times Cause_1 + \beta_2 \times Cause_2 + \cdots + \beta_n \times Cause_n \quad (99)$$

The other commonly used Type II model is the Proportional Hazards Model (PHM). This model takes the environmental conditions (termed z_j) into account to modify the baseline hazard rate $\lambda_0(t)$ to produce a new hazard rate:

$$\lambda(t; z) = \lambda_0(t) e^{(\sum_{j=1}^{q} \beta_j z_j)} \quad (100)$$

The term z_j is a multiplicative factor, an explanatory variable or covariance that explains the effect on failure rate. The parameter $\lambda_0(t)$ is an arbitrary baseline hazard function and β_j is a model parameter (Eklund 2009).

As an example, the proportional hazard model is a type II prognostic model. Using the disk drive example, one can determine the expected failure rate if the disk drive's total operating hours to date and the disk drive's prior operating condition such as historical temperatures and number of disk accesses are known. These models are usually cause and effect based.

Type III prognostic methods are condition-based, namely they characterize the lifetime of a system in operation in its specific environment. They estimate the remaining life of a specific component or the entire system. Among methods used in Type III prognostics are the General Path Model (GPM), Neural Network models, Expert systems, Fuzzy rule-based systems, and multi-state analysis. Another example of type III model is the cumulative Damage model.

One of the prognostic models that can gather and learn failure parameter data is Artificial Neural Networks (ANN) and is discussed next in this paper. As sensors become smaller and smarter, the proliferation of sensors implies increasing volume of data that can be processed for prognostics. ANNs are ideal constructs for medical prognostics because they can model feed-forward systems, compute non-linear relationships and analyze very large number of input data. In fact, given their parallel architecture, ANNs can function or

even substitute for missing data. They are able to learn the inherent rules in a given system, can maintain long term memory and discern patterns even in noisy and changing environments. Because of these characteristics, ANNs are increasingly selected for prognostics studies and this is a reason that I've selected ANNs for this research.

The cumulative damage model tracks the irreversible accumulation of damage in systems or components. The statistical cumulative damage model considers the number of possible damage states and a transition matrix (for representing a multi-state Markov Chain) to provide a damage prediction for multiple cyclical loads.

Another type III method is the Shock model. This approach is used to predict the RUL for a system subject to randomly arriving shocks. The shocks deliver certain damage of random magnitude to the system. These models are continuous in time and consequently, the degradation measures are also continuous. Shock models are estimated from historical failure data. They are similar to the Markov Chain model, except that the time between shocks and the shock magnitudes are continuous and random variables.

GPM models were proposed in 1993 as a statistical method for estimating a time-to-failure distribution using degradation measures. GPM models assume that the degradation of a system is a function of time, duty cycle or some other measure. The model extrapolates a degradation function to predict RUL. GPM makes two assumptions: A) Each individual device (or system) has a unique degradation signal and B) The failure occurs at a critical threshold. This model starts with a parametric model to the exemplar degradation paths. It then computes the mean and covariance values to explain individual random parameters. It can use Bayesian probability functions to modify the posterior RUL values from apriori data. It extrapolates the critical failure threshold to estimate RUL.

The reliability and survival analysis techniques, both parametric and non-parametric methods are noteworthy of discussion. These methods as will be discussed later can by applied to prognostics. Finally, there is the Artificial Neural Network (ANN) models that are used for this research and will be explained in more detail later in this paper. A synthesis of how these methods compare and their suitability to predicting clinical patient status will be presented.

Reliability and Predictive Analytics:

Reliability is defined as the probability that product will perform its intended function, satisfactorily for its intended life when operating under specified condition (Kapur 2010). A clinical definition can be derived from this technical description; Reliability is the probability that a patient will not develop certain medical complication during the length stay under medical care of the care provider(s). Reliability is measured by several indicators such as Mean Time Between Failure (MTBF), Failure rate and Percentiles of Life. Each measurement can be computed from corresponding equations that are derived from empirical and statistical distribution functions.

In general, Reliability at time t, is shown as *R(t)*. Failures are measured by *f(t)*, the probability density function for the time of failure, a random variable T. The cumulative distribution function for random variable *T* is shown by *F(t)*. Thus, one can write the following expressions to define *F(t)*, *f(t)* and *R(t)*:

$$F(t) = P\,[T \leq t] = \int_0^t f(\tau)d\tau \tag{101}$$

$$f(t) = \frac{dF(t)}{dt} \tag{102}$$

$$R(t) = 1 - F(t) \tag{103}$$

Additional treatment of Reliability can be found from text by Kapur and Lamberson (Kapur, Lamberson 1977).

Appendix B: A Neural Network Example

Consider the following classification problem as described in section 2. Suppose we're considering classifying patients by only four input variables, Glucose (G), Body mass (M), Systolic Blood pressure (S) and Platelet count (P):

Values for G, M, S, and P for past patients are given for the model to train on, as listed below. The first step in classification is to normalize the input data to unit length. Normalizing to unit length implies that the sum of squares of values in a given data set are equal to 1. This technique was explained in section 4.2.

The classification problem is defined as follows. There are two sets of data vectors: one set of data vectors belong to the TRUE set (Patients with DVT) and another set belongs to the FALSE set (No DVT cases). A new patient with the normalized data is introduced and the goal is to classify that patient with the normalized values of: [0.75, 0.32, 0.60, 0.21].
The previously classified data sets and their corresponding new normalized values are computed as follows:

[117, 194, 140, 276] , DVT = TRUE, normalized: [0.31, 0.51, 0.37, 0.72]
[120, 164, 213, 315] , DVT = TRUE, normalized: [0.27, 0.38, 0.49, 0.73]
[115, 145, 170, 288] , DVT = TRUE, normalized: [0.30, 0.38, 0.44, 0.75]
[122, 165, 155, 290] , DVT = TRUE, normalized: [0.31, 0.43, 0.40, 0.75]

For patients with no DVT outcome:

[122, 144, 110, 236] , DVT = FALSE, normalized: [0.38, 0.45, 0.34, 0.73]
[140, 154, 153, 176] , DVT = FALSE, normalized: [0.45, 0.49, 0.49, 0.56]
[145, 135, 130, 218] , DVT = FALSE, normalized: [0.45, 0.42, 0.40, 0.68]
[132, 155, 115, 190] , DVT = FALSE, normalized: [0.44, 0.51, 0.38, 0.63]

The following kernel for computing PNN pattern and summation based on derivations from Equations (39) and (41):

$$z_A = f_A(x) = \sum_{i=1}^{N_k} e^{\frac{(x^t w_{ki} - 1)}{\sigma^2}}$$

The new patient with the normalized data set of [0.75, 0.32, 0.60, 0.21] is to be classified. In this example, it's assumed that σ is equal to 1.0 for sake of simplifying calculations. Next, it's possible to compute $f_A(x)$ for each data set:

.31*.75+.51*.32+.37*.60+.72*.21 – 1.0 = -0.239 exp (-0.239) = 0.787

exp (-0.220) = 0.803

exp (-0.228) = 0.796

exp (-0.231) = 0.794

Sum1 = 3.180

Similarly for the second class, it's possible to compute:

.38*.75+.45*.32+.34*.60+.73*.21 – 1.0 = -0.213 exp (-0.213) = 0.808

exp (-0.095) = 0.910

exp (-0.144) = 0.866

exp (-0.145) = 0.865

Sum2 = 3.449

Since Sum2 > Sum1, it implies that the data points for this patient are closer to FALSE classification as the sum of values associated with the FALSE class is higher. Thus this patient belongs to the FALSE classification and is predicted to have DVT=FALSE (i.e. prognostics for DVT is negative).

Appendix C: Back Propagation Algorithm Derivation

In general, there are two types of learning processes: the batch mode and the sequential mode. On the conditions for these mathematical calculations is that the function $\varphi_j\left(v_j(n)\right)$ must be continuous and differentiable so one can obtain $\varphi_j'\left(v_j(n)\right)$. Also the choice of η as the learning rate is important to be set at the appropriate value. The backward propagation is an approximation to the steepest descent algorithm. If the researcher uses a small η there is risk of getting unstable results. Rumelhart (Rumelhart 1986) suggested adding a momentum value to the learning rate η according to this algorithm:

Case 1: If $\frac{\partial E(n)}{\partial w_{ji}(n)}$ has the same sign for all n, then one can say that $|\Delta w_{kj}|$ grows in magnitude. This represents the case of an accelerated descent.

Case 2: If in contrast $\frac{\partial E(n)}{\partial w_{ji}(n)}$ alternates its sign in every iteration, then $|\Delta w_{kj}|$ is small in magnitude and more controllable than the first case. The momentum term is significant because it accelerates learning but also help to avoid local optima.

The Back propagation algorithm employs the following eight steps. This is adapted from Zurada (Zurada 1997) and Sengupta (Sengupta 2009):

Given *p* training pairs shown by: $\{z_1, d_1, z_2, d_2, \ldots, z_P, d_P\}$,

where z_i is a ($I \times 1$), d_i is ($K \times 1$), and $i = 1, 2, \ldots, P$.

Step 1: Choose > 0, E_{max} . Weights W and V are initialized at small random values. W is ($K \times J$), V is ($J \times 1$). Initialize q, p and E:

$$q \leftarrow 1, p \leftarrow 1, E \leftarrow 0$$

Step 2: Start training. Input is presented and the layers' outputs are computed using $\Delta w_i = cf(w_i^t\, x)x$:

$z \leftarrow z_p, \qquad d \leftarrow d_p$

$y_j \leftarrow f\left(V_j^t z\right), for\ j = 1, \ldots, J$ Where V_j is a column vector and is the *j*th row of V, and

$o_k \leftarrow f(W_k^t y), for\ k = 1, \ldots, K$ Where W_k is a column vector and is the *k*th row of W.

Step 3: Compute the Error value by comparing the desired output versus network output:

$E \leftarrow E + \frac{1}{2}\,(d_k - o_k)^2\ for\ k = 1, \ldots, K.$

Step 4: Compute error signal vectors δ_o and δ_y of both layers. Vector δ_o is (K X 1) and δ_y is (J X 1). The error signal terms of the output layer in this step are:

$$\delta_{ok} = \frac{1}{2}\left[(d_k - o_k)(1 - \delta_k^2)\right] \quad for\ k = 1, \ldots, K$$

The error signal terms of the hidden layer in this step are computed by:

$$\delta_{yj} = \frac{1}{2}(1 - y_j^2)\sum_{k=1}^{K}\delta_{ok} w_{kj}\ , for\ j=1,\ \ldots,\ J$$

Step 5: Adjust the output layer weights by:

$$w_{kj} = w_{kj} + \eta\delta_{ok} y_j\ for\ k = 1, \ldots, K\ and\ j = 1, \ldots, J$$

Step 6: Adjust the hidden layer weights:

$$V_{ji} = V_{ji} + \eta\delta_{yj} z_i\ for\ i = 1, \ldots, I\ and\ j = 1, \ldots, J$$

Step 7: If $p < P$ then increment p and q: $q \leftarrow q + 1, p \leftarrow p + 1$, go to step 2, otherwise go to step 8.

Step 8: The training cycle is completed. If $E < E_{max}$ the training session is finished. The weights are defined by W, V, q and E. If $E > E_{max}$ then $p \leftarrow 1, E \leftarrow 0$ and initiate a new training cycle by going to step 2.

Most recently, practitioners prefer other algorithms such as the Conjugate Gradient Descent and Simulated Annealing (Masters 2005) to backward propagation. These algorithms offer faster convergence to learning. Conjugate Gradient Descent is a deterministic optimization method that attempts to find local minimum of a function. Simulated Annealing is designed to ensure that the training algorithm overcomes getting trapped in local minima.

Appendix D: NeuroSolutions Software Description

In this research, four different commercial software packages for neural network modeling were evaluated. The evaluation considered several criteria for selection. The packages were compared on the following criteria:

- A large library of ANN algorithms
- Ability to combine multiple algorithms
- Robustness of the software on large data sets so it would be stable and not crash
- Ability to measure accuracy and provide accuracy measurements for each model
- Ability to use Excel input data files
- Ability to export (or package) the model for external use

This evaluation included building actual models using the DVT/PE case study and learning the pros and cons of each package. Of the four models evaluated, a commercial neural network software entitled NeuroSolutions™ was employed for building all ANN models. NeuroSolutions is an object-oriented environment for neural networks that implements both static and dynamic, arbitrary, user-defined topologies, which can be adapted with the most popular learning paradigms. NeuroSolutions is developed, supported and licensed by NeuroDimension, Incorporated, Gainesville, Florida.

The graphical user interface uses the electronic design metaphor for neural network design, in which neural "components" are placed on a "breadboard" and interconnected with each other. The component icons are associated univocally with the objects that implement the functionality of the package. The interface is built with the idea of showing to the user all the important variables that pertain to the configuration of the components.

Adaptive systems can be better understood if all the internal variables and parameters can be visualized. For this reason, NeuroSolutions includes an extensive set of graphical probes. Figure 42 shows an example of a NeuroSolutions neural network (breadboard) model which includes a probe showing the cost (Mean Square Error, MSE) over time and one that shows the network output versus the desired output for a time series dataset.

NeuroSolutions can implement a wide array of models including linear regression models, nonlinear regression models, multilayer perceptrons (MLPs) with arbitrary topologies, radial basis function networks, digital filters, time lagged networks of arbitrary topologies, recurrent networks of arbitrary topologies, associative memories, support vector

machines, neuro-fuzzy, principal (and nonlinear) component networks, competitive and Kohonen networks. It implements several types of backpropagation to train static and recurrent networks (static backpropagation, fixed point learning, backpropagation thought time), several cost functions (L1, L2, Linf norms). It implements several types of search procedures (momentum learning, delta-bar-delta, conjugate gradient and Fahlman's quickprop), and enables the mixing of any one of these models.

The package is a very open development environment in that it enables the user to extend the basic components by modifying the default C code and compiling the new components as DLLs (Dynamic Link Libraries).

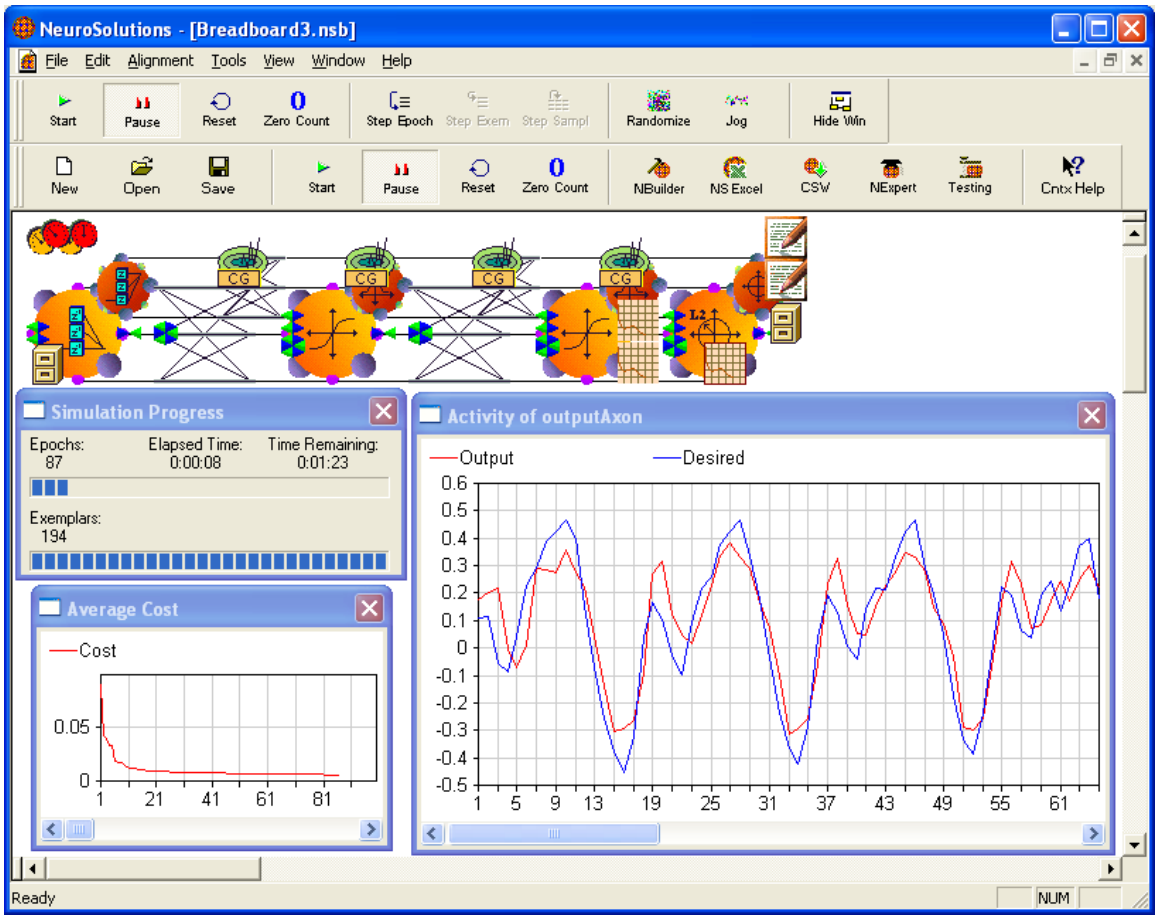


Figure 42: NeuroSolutions' object oriented graphical environment

NeuroSolutions for Neural Network Control

NeuroSolutions is an ideal environment for developing neural control systems for a number of reasons. Its icon-based graphical user interface and extensive probing capabilities provides a rapid prototyping environment for experimenting with various neural network

architectures and algorithms. If a desired algorithm is not included with the package, then a custom one can be easily implemented by writing Dynamic Link Libraries (DLLs) to override the functionality of the default neural components.

NeuroSolutions has an extensive API library (Application Programming Interface) that makes it fully accessible from external programs. Once the neural network is designed, an automated test procedure could be written to feed the neural network the input data and extract the output data. This data could reside on the local file system or it could come from an external source which is interfaced to one of the computer's I/O ports.

The final stage of building the control system is to deploy the neural network to the actual control environment. NeuroSolutions allows you to encapsulate a specific neural network design by automatically generating the core (non-graphical) C++ source code for it. This C++ code can then be embedded into the code for the digital control program or compiled as a self-contained object such as a dynamic link library (DLL).

NeuroSolutions currently implements an impressive array of first order search techniques (gradient, momentum, adaptive step sizes (two different methods), annealed step size, independent step size) and an optimal line search algorithm called – Conjugate Gradient.

NeuroDimension, Inc. was founded in 1991, in Gainesville, Florida, by medical and computer engineering specialists whose original goal was to develop software tools that would improve their own efficiency and productivity. NeuroDimension's first product, NeuroSolutions v1.0, was released in 1994. Since that time there have been numerous enhancements to NeuroSolutions and several new products have been released (NeuroSolutions for Excel, Custom Solution Wizard, Genetic Server, and TradingSolutions), resulting in continuous sales and personnel growth.

Appendix E: The Oracle Program

The oracle program is the overseer module that selects the Ensemble with the highest value of AUC. The oracle program was written in R, an open source statistical package (Wang 2012) and was tested on the 1,073 data sets and all Ensembles. The source codes is listed below. The program computes the AUC for each ANN algorithm and Ensembles compared to the actual truth (presence or absence of disease) for prior data. The first column is the actual truth and the subsequent columns are the data for ANN algorithms and Ensembles.

```
#####################################
###### R code for computing AUC ######
###### Peter Ghavami – Adapted  ######
######      From Wang 2012      ######
#####################################

### Load library packages ###

install.packages("verification")
library(verification)

### Import the input data file ###

## select the data file in the pop out window
data<-read.csv(file.choose(),header=T)  # Import data from csv file

### Purpose: write a R function to compute AUC ###
### function name: getAUC
### function input:
###      D: true disease status
###      T: continuous test results of ANN and Ensembles
### function output: AUC value for this test

getAUC<-function(D, T){
roc.area(D, T)$A
}                      # Call the getAUC function.

### compute the AUC values for each test in the data

### result for MLP
getAUC(data[,1],data[,2])      #MLP data is in the 2nd column

### result for PNN
getAUC(data[,1],data[,6])      #PNN data is in the 6th column

### result for GFF
getAUC(data[,1],data[,10])     #GFF data is in the 10th column

### result for SVM
getAUC(data[,1],data[,14])     #SVM data is in the 14th column

### result for Ensemble1
```

```
getAUC(data[,1],data[,18])     #Ensemble1 data is in the 18th column

### result for Ensemble2
getAUC(data[,1],data[,22])     #Ensemble2 data is in the 22nd column

### result for Ensemble3
getAUC(data[,1],data[,26])     #Ensemble3 data is in the 22nd column

### result for Ensemble4
getAUC(data[,1],data[,30])     #Ensemble4 data is in the 30th column

### result for Ensemble5
getAUC(data[,1],data[,34])     #Ensemble5 data is in the 34th column
```

REFERENCES:

(Aamodt, Plaza 1994). Aamodt, A., Plaza, E., Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches , *AI Communications, Vol, 7, Nr. 1*, March 1994.

(Adams, Wert 2005) Adams, J. B., Wert, Y., "Logistic and Neural Network Models for Predicting a Hospital Admission", Journal of Applied Statistics, Vol. 32, No. 8, 861-869, 2005

(Allison 1995). Allison, P. D., *Survival Analysis using the SAS system*, SAS Institute publication, 1995, Cary, NC.

(AMA 2010). CPT 2011 Professional Edition, Michelle Abraham, *American Medical Association*, American Medical Association Press, Oct 20, 2010.

(Arthi, Tamilarasi 2008). Arthi, K., Tamilarasi, A., Prediction of autistic disorder using neuro fuzzy systems by applying ANN technique, *International Journal of Developmental Neuroscience, 26 (2008)* 699-704

(Baxt 1991). Baxt, W. G., *Use of an Artificial Neural Network for the Diagnosis of Myocardial Infarction*. Annals of Internal Medicine, Dec 1, 1991, Vol. 115, no. 11, pg 843-848

(Bewick, Cheek, Ball 2004) Bewick, V., Cheek, L., Ball, J., "Statistics review 13: Receiver operating characteristic curves", *Critical Care*, Vol. 8, no. 6, December 2004

(Blount, Ebling, Eklund, James, et al. 2010) Blount, M., Ebling, M. R., Eklund, J. M. , James, A. G. , McGregor, C ., Percival, N. , Smith, K. P., and Sow, D. (2010). Real-time Analysis for Intensive Care. Development and Deployment of the Artemis Analytic System. *IEEE Engineering in Medicine and Biology Magazine,* March/April 2010

(Bourdes, et al 2011) Bourdes, V., Ferrieres, J., Amar, J., Amelineau, E., Bonnevay, S., Berlion, M., Danchin, N., "Prediction of persistence of combined evidence-based cardiovascular medications in patients with acute coronary syndrome after hospital discharge using neural networks", *Medical & Biological Engineering Computing*, 49:947-955, 2011

(Bottaci, Drew, Hartley, Hadfield, et al. 1997). Bottaci, L., Drew, P. J., Hartley, J. E., Hadfield, M. B., Farouk, R., Lee, P. WR., Macintyre, I. MC., Duthie, G. S., Monson, J. RT. (1996). Artificial Neural Networks Applied to Outcome Prediction for Colorectal Cancer Patients in Separate Institutions. *The Lancet, Vol. 350, Issue 9076*, Aug 16, 1997, Pg 469-472

(Breiman, Friedman, Olshen, Stone 1984) L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, *Classification and regression trees*. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software, 1984.

(Brown, Strong, 2001). Brown, S.W., Strong, V., The use of seizure-alert dogs, *Seizure*, *2001, 10*:39-41.

(CEBM 2012). Center for Evidence Based Medicine website, EBM Tools, http://www.cebm.net/index.aspx?o=1023 , accessed, March 22, 2012

(Coble, Hines 2009). Coble, J., Hines, J. W., Identifying Optimal Prognostic Parameters from Data: A Genetic Algorithms Approach, *Annual Conference of the Prognostics and Health Management Society*, 2009

(Coble, Hines 2009). Coble, J., Hines, J. W., Fusing Data Sources for Optimal Prognostic Parameter Selection, *Sixth American Nuclear Society International Topical Meeting on Nuclear Plant Instrumentation, Control, and Human-Machine Interface Technologies NPIC & HMIT 2009*, Knoxville, Tennessee, April 5-9, 2009

(Collett 1994). Collett, D., *Modeling survival data in medical research*. London: Chapman & Hall.

(Daley, Narayanan, Leffler 2010). Daley, M., Narayanan, N., Leffler, C. W., Model-derived assessment of cerebrovascular resistance and cerebral blood flow following traumatic brain injury, *Experimental Biology and Medicine, Vol 235*, April 2010

(Davenport, Dennis, Wellwood, Warlow 1996) Davenport, R.J., Dennis, M.S., Wellwood, I., Warlow, C., "Complications after acute stroke", *Stroke*, Vo. 27, pg 415-420, 1996

(Dayhoff, DeLeo 2001). Dayhoff, J. E., DeLeo, J. M., Artificial Neural Networks, Opening the Black Box. *Cancer 2001; 19*:1615-35. Presented at the Conference on Prognostic Factors and Staging in Cancer Management: Contributions of Artificial Neural Networks

and Other Statistical Methods.

(Delen 2009) Delen, D., "Analysis of cancer data: a data mining approach", *Expert Systems*, February 2009, Vol. 26, No. 1

(Dictionary.com 2012). www.Dictionary.com, online, an IAC company, Accessed, Jan 2012.

(Doyle, Francis, Tannenbaum 1990). Doyle, J., Francis, B., Tannenbaum, A., *Feedback Control Theory*, Macmillan Publishing Co, 1990.

(Dybowski, Grant, Weller, Chang 1996). Dybowski, R., Gant, V., Weller, P., and Chang, R., Prediction of Outcome in Critically ill Patients Using Artificial Neural Network Synthesised by Genetic Algorithm. *The Lancet, Vol 347, Issue 9009*, April 27, 1996, pg 1146-1150

(Eklund 2009). Eklund, N. H.W., Prognostics and Health Management - Part 1: Data Driven Anomaly Detection & Diagnosis, *Annual Conference of the Prognostics and Health Management Society*, Diagnostics Tutorials, 2009

(Floyd, Lo Yun, Sullivan, et al 1994). Floyd, C. E., Lo, J. Y., Yun, A. J., Sullivan, D. C., Kornguth, P. J., Prediction of Breast Cancer Malignancy Using an Artificial Neural Network. *Cancer, Vol. 74, no. 11*, Dec. 1, 1994.

(Fuller, McCullough, Bao 2009). Fuller, R. L., McCullough, E. C., Bao, M. Z., Averill, R. F., "Estimating the Costs of Potentially Preventable Hospital Acquired Complications", *Healthcare Financing Review*, Vo. 30, No. 4, Summer 2009

(Gao, Young, Ornstein, Pile-Spellman, et al. 1997). Gao, E., Young W., Ornstein, E., Pile-Spellman, J., Qiyuan, M., A theoretical model of cerebral hemodynamics: Application to the study of Arteriovenous Malformations, *Journal of Cerebral Blood Flow and Metabolism, 1997, 17*, 905-918

(Ghavami, Kapur 2011) P. Ghavami, K. Kapur, "Prognostics & Artificial Neural Network Applications in Patient Healthcare", *Proceedings of IEEE Prognostics and Health Management Conference*, June 2011

(Graunt 1662). Graunt, J., Natural and Political Observations Made Upon the Bills of Mortality.

(Hahnfeldt, Panigraphy, Folkman, Hlatkey, et al. 1999). Hahnfeldt, P., Panigraphy, D., Folkman, J., Hlatkey, L., Tumor development under angiogenic signaling: a dynamic theory of tumor growth, treatment response and postvascular dormacy, *Cancer Research 59*, 4770-4778, 1999

(Haykin 1998). Haykin, S., *Neural Networks, A Comprehensive Foundation*, 2nd Edition, Prentice Hall, 1999

(Hines 2009). Hines W. J., Empirical Methods for Process and Equipment Prognostics, *Annual Conference of the Prognostics and Health Management Society*, Prognostics Tutorials, 2009

(Hu, Youn, Wang 2010). Hu, C., Youn, B.D., Wang, P., Ensemble of data-driven prognostics algorithms with Weight Optimization and K-Fold Cross Validation, *Annual Conference of the Prognostics and Health Management (PHM) Society*, Oct 10-16 2010, Portland, OR.

(INCOSE 2000). What is a system?, Version 2.0, *INCOSE (International Council on Systems Engineering Council) Systems Engineering Handbook*, July 2000.

(Jervis, McGinn 2008) Jervis, R., McGinn, T., Evidence-based Medicine, Clinical prediction rules for hospitals, *Mount Sinai Journal of Medicine* 75: 472-477, 2008

(Kapur 2010). Kapur, K., *Seminar on Prognostics, Dept. of Industrial & Systems Engineering, University of Washington*, Feb-March, 2010.

(Kapur, Lamberson 1997). Kapur, K., Lamberson, L. R., *Reliability in Engineering Design*, 1977

(Kimmel, Axelrod 2002). Kimmel, M., Axelrod, D.E., *Branching processes in biology*, Springer Verlag, New York, NY, 2002

(Kirton, Winter, Wirrell, Snead 2008) Kirton, A., Winter, A., Wirrell, E., Snead, O. C., "Seizure response dogs: Evaluation of a formal training program*", Epilepsy & Behavior*, 13 (2008) 499-504.

(Kodell, Pearce, Baek, et al. 2009). Kodell, R. L., Pearce, B. A., Baek, S., Moon, H., Ahn, H., A model-free ensemble method for class prediction with application to biomedical decision making, *Artificial Intelligence in Medicine (2009), 46*, 267-276

(Kon, Plaskota 2000). Kon, A., M., Plaskota, L., Complexity of Predictive Neural Networks, *International Conference on Complex Systems*, May, 2000.

(Kwakernaak, Sivan 1972). Kwakernaak, H., Sivan, R., *Linear Optimal Control Systems*, John Wiley & Sons, 1972

(Laupacis, Sekar, Stiell 1997) Laupacis, A., Sekar, N., Stiell, I. G., Clinical prediction rules. A review and suggested modifications of methodological standards. *JAMA* 1997; 277:488-94.

(Ling, Huang, Zhang 2003). Ling, C.X., Huang, J., Zhang, H., "AUC: a Statistically Consistent and more Discriminating Measure than Accuracy", *International Joint Conference on Artificial Intelligence, 2003, Vol. 18*, pages 519-526, Lawrence Erlbaum Associates, LTD.

(Ling, Huang, Zhang, 2003). Ling, C.X., Huang, J., Zhang, H., "AUC: A Better Measure than Accuracy in Comparing Learning Algorithms", *Lecture Notes in Computer Science*, 2003, ISSU 2671, pages 329-341, Springer-Verlag

(Limaye, Mastrangelo, Zerr, Jeffries 2008). Limaye, S. S., Mastrangelo, C. M., Zerr, D. M., Jeffries, H., A statistical approach to reduce hospital-associated infections, *Quality Engineering, 20*:414-425, 2008

(Linder, Geier, Kolliker 2004) Linder, R. Geier, J., Kolliker,M., "Artificial neural networks, classification trees, and regression: Which method for which customer base?" *Database Marketing & Customer Strategy Management*, Vol 11, 4, 344-356, 2004

(Lisboa, Taktak 2005). Lisboa, P. J., Taktak, A. F.G., The Use of Artificial Networks in Decision Support in Cancer: A Systematic Review. *Neural Networks, Vol 19, Issue 4*, May 2006, pg 408-415

(Macal 2005). Macal, C., Model Verification and Validation, The University of Chicago and Argonne National Laboratory, *Workshop on "Threat Anticipation: Social Science Methods and Models"*, April 7-9, 2005, Chicago, IL.

(Maguire 2007) Maguire, P., "The new crackdown on preventable complications", *Today's Hospitalist*, October 2007.

(Masters 1995). Masters, T., *Advanced Algorithms for Neural Networks: A C++ Sourcebook*, Wiley, New York, 1995

(McGinn, Guyatt, Wyer, Naylor, et al. 2000). McGinn, T. G., Guyatt, G. H., Wyer, P. C., Naylor, C. D., Stiell, I. G., Richardson, W. S., Users' Guide to Medical Literature, *JAMA 2000; 284(1)*:79-84; For the Evidence-based Medicine Working Group.

(Merriam-Webster Dictionary 2011). Merriam-Webster dictionary online, www.Merriam-webster.com/dictionary/, an Encyclopedia Britannica Company. Accessed December 2011.

(MIT 2010) See http://classics.Mit.edu/Hippocrates/ prognost.html. Date accessed: Feb 2010.

(Monterola, Lim, Garcia, Saloma 2002). Monterola, C., Lim, M., Garcia, J., Saloma, C., Feasibility of a Neural Network as Classifier of Undecided Respondents in a Public Opinion Survey, *International Journal of Public Opinion Research, Vol. 14, No. 2*, 2002

(Neuro Dimenstions, Inc 2011). NeuroDimension, Inc., Gainesville, Florida, NeuroSolutions software, Version 6.0

(Niu, Yang, Pecht 2010). Niu, G., Yang, B., Pecht, M., Development of an optimized condition-based maintenance system by data fusion and reliability-centered maintenance, *Reliability Engineering and System Safety*, V95, n7, p786-796, 2010

(O'Connor, Bennett, Stacey, Barry, et al. 2009). O'Connor, A. M., Bennett, C.L., Stacey, D., Barry, M., Col, N. F., Eden, K.B., Entwistle, V. A., Fiset, V., Decision aids for people facing health treatment or screening decisions (Review), *The Cochrane Collaboration*, Wiley 2009

(Ozbay 1999). Ozbay, H., *Introduction to Feedback Control Theory*, CRC Press, 1999

(Park, Kim, Chun 2006). Park, Y., Kim, B., Chun, S., "New knowledge extraction technique using probability for case-based reasoning: application to medical diagnosis*, Expert Systems*, Feb 2006, Vol. 23, No. 1

(Pecht 2008). Pecht, M., *Prognostics and Health Management of Electronics*, Wiley 2008

(Peysson, Ouladsine, Outbib 2009). Peysson, F., Ouladsine, M. and Outbib R., Complex System Prognostics: A New Systemic Approach, *Annual Conference of the Prognostics and Health Management Society*, 2009

(Principe, Euliano, Lefebvre 1999). Principe, J. C., Euliano, N.R., Lefebvre, W.C., *Neural and Adaptive Systems, Fundamentals Through Simulations*, John Wiley & Sons, 1999

(Principe 2011). Principe, J. C., Conversations with Jose` C. Principe, University of Texas, Sept. 2011

(Prodormidis, Chan, Stolfo 2000). Prodormidis, A.L., Chan, P.K., Stolfo, S.J., Meta-learning in distributed data mining systems: Issues and Approaches, *Advances in Distributed Data Mining*, MIT Press, 2000

(Ravdin, Clark, 1992). Ravdin, P. M. and Clark, G. M., A Practical Application of Neural Network Analysis for Predicting Outcome of Individual Breast Cancer Patients. *Breast Cancer Research and Treatment, Vol. 22, No. 3*, Oct. 1992. pg 285-293

(Rumelhart, Hinton, Williams 1986). Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986) Learning representations by back-propagating errors, *Nature*, *vol. 323*, pp. 533-536, 1986

(Schlimmer, Granger, Jr. 1986). Schlimmer, J.C., Granger, Jr., R.H., Incremental Learning from Noisy Data, *Machine Learning 1*:317-354, 1986, Kluwer Publishers, Boston

(Sengupta 2009). Sengupta, S., *Lecture series on Neural Networks and Applications by Prof. S. Sengupta*, Department of Electronics and Electrical Communication Engineering, Indian Institute of Technology, Kharagpur, source: NPTEL, http://nptel.iitm.ac.in , accessed 2009- 2012

(Smye, Clayton 2002). Smye, S. W., Clayton, R. H., Mathematical modeling for the new millennium: medicine by numbers, *Medical Engineering & Physics, 24 (2002)*, 565-574

(Souter 2011). Souter, M., Conversations on diagnostic markers and predictors, April 7, 2011.

(Spiegelman, Schneeweiss, McDermott, 1996). Spiegelman, D., Schneeweiss, S., McDermott, A., Measurement Error Correction for Logistic Regression Models with an "Alloyed Gold Standard", *American Journal of Epidemiology*, Vol. 145, no. 2, 1996

(Spruance, Reid, Grace, Samore 2004) Spruance, S. L., Reid, J. E., Grace, M., Samore, M., "Hazard Ratio in Clinical Trials", *Antimicrobial Agents and Chemotherapy*, Vol. 48(8), Aug 2004

(StopDVT.org 2011). www.StopDVT.org website, page accessed: http://stopdvt.org/FAQ.aspx, Accessed October 2011.

(Strong, Brown Walker 1999) Strong, V., Brown, S.W., Walker, R., "Seizure-alert dogs-fact or fiction"?, *Seizure*. 1999; 8:26-65.

(Swierniak, Kimmel, Smieja 2009)Andrezej Swierniak, Marek Kimmel, Jaroslaw Smieja, "Mathematical modeling as a tool for planning anticancer therapy", *European Journal of Pharmacology*, 625 (2009) 108-121

(Toll, Janssen, Vergouwe, Moons 2008). Toll, D. B., Janssen, K. J. M., Vergouwe, Y., Moons, K. G. M., Validation, updating and impact of clinical prediction rules: A review. *Journal of Clinical Epidemiology*, 61 (2008) 1085-1094.

(Tsai, Pollock, Brownie 2009). Tsai, K., Pollock, K., Brownie, C., "Effects of violation of assumptions for survival analysis methods in radiotelemetry studies", *Journal of Wildlife Management*, 63(4):1369-1375, 2009

(TU 1996). TU, J.V., Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes, *Journal of Clinical Epidemiology*, 1996, Nov; 49(11): 1225-31.

(Uckun, Goebel, Lucas 2008). Uckun, S., Goebel, K. and Lucas, P. J. F., Standardizing Research Methods for Prognostics, *2008 International Conference on Prognostics and Health Management,* 2008

(Vichare, Pecht 2006). Vichare, N. M., and Pecht, M., Prognostics and Health Management of Electronics, *IEEE Transactions on Components and Packaging Technologies, Vol 29, No. 1*, March 2006.

(Virchow 1856). Virchow, R.,Virchow's Triad. Virchow's Triad was first formulated by the German physician Rudolf Virchow in 1856.

(Wang 2012). Wang, Z., Conversations about neural network algorithms and accuracy measures. Department of Biostatistics, University of Washington.

(Wang 2012). Wang, Z., Conversations and collaboration for calculating AUC using R statistical language. Department of Biostatistics, University of Washington.

(Webber, Litt, Wilson, Lesser 1994). Webber, W. R. S., Litt, B., Wilson, K., Lesser, R. P. (1994), Practical detection of epileptiform discharges (EDs) in the EEG using an artificial neural network: a comparison of raw and parameterized EEG data. *Electroencephalography and clinical Neurophysiology, 91,* 194-204

(Wells, Anderson, Bromanis, Mitchell, et al. 1997). Wells, P.S., Anderson, D.R., Bromanis, J., Guy, F., Mitchell, M., Gray, L., Clement, C., Robinson, K.S., Lewandowski, B., Value of assessment of pretest probability of deep-vein thrombosis in clinical management, *The Lancet, Vol 350, Issue 9094*, pg 1795-1798, 20 December 1997.

(WHO 2012) *Library of ICD9 and ICD10 codes*, World Health Organization's library of International Statistical Classification of Diseases and Related Health Problems, http://www.who.int/classifications/icd/revision/en/index.html, accessed, March 09, 2012

(Williamoski, Chen 1999). Williamowski, B. M., Chen, Y., Efficient algorithm for Training Neural Networks with one Hidden Layer, *IEEE International Joint Conference on Neural Networks*, 1999

(Williams, Pembroke 1989). Williams, H., Pembroke, A., Sniffer dogs in the melanoma clinic?, *Lancet, 1989, 1(8640)*;734

(Wishart 1969). Wishart, D., Symposium on Control Theory, A survey of Control Theory. *Journal of the Royal Statistical Society, Series A*, Royal Statistical Society, 1969.

(Yu, Liu, McKenna, Reisner, et al. 2006). Yu, C., Liu, Z., McKenna, T., Reisner, A. T., Reifman, J., A Method for Automatic Identification of Reliable Heart Rates Calculated from ECG and PPG Waveforms, *Journal of the American Medical Informatics Association, vol. 13, No. 3*, May/Jun 2006.

(Zadeh, Desoer 1963). Zadeh, L. A., and Desoer, C., *Linear Control Theory*, Springer-Verlag, 1963

(Zurada 1997). Zurada, J. M., *Introduction to Artificial Neural Network*, Jaico Publishing House, Second Edition, 1997

VITA

Peter K. Ghavami received his BA from Oregon University in Mathematics Sciences with emphasis in Computer Science. He received his M.S. in Engineering Management from Portland State University. His career started as software engineer, with progressively responsibilities as IT consultant at IBM Corp., Director of engineering and VP of Product Development at various high technology firms. He is currently Director of Imaging Informatics at UW Medicine - Harborview Medical Center and working towards completing his PhD in Industrial and Systems Engineering at University of Washington. He has authored papers on software process improvement, vector processing, distributed network architectures, and software quality. He authored the book *Lean, Agile and Six Sigma Information Technology Management* in 2008. He is a member of IEEE Reliability Society, IEEE Life Sciences Initiative and Engineers without Borders.