Variable importance in artificial neural networks with real and simulated data

Diego Ramírez Vargas
18043097
June 8, 2019

Contents

List of Figures

List of Tables

Abstract

Artificial neural networks (ANNs) are known to be complex methods. These intricate networks are undoubtedly "black box" models. This means they are powerful in predicting the results of real-world problems, but it is hard to understand them. These networks are more reliable for prediction than for explaining the relationships in the data. Moreover, it is arduous to observe how the variables affect the model. Additionally, the optimum number of hidden neurons and layers is unknown.

Moreover, to determine how the number of hidden neurons and layers affect the outcome the metric MAPE and 50 ANNs were employed. Additionally, using simulation, three key quantities are progressively changed. These are the correlation, the sample size and the mislabeling probability in the data set.

This work was first motivated by an application of neural networks to breast cancer data. Then to understand the results of the methods applied to this type of data simulated data was employed.

1 Introduction

In this section the concept of "Black box" and specific problems in ANNs are introduced. Also, information about the breast cancer is given. The goals of the project and contributions of the dissertation to the knowledge of Artificial Neural Networks are stated. Additionally, the types of ANNs employed in Breast Cancer. It is divided in six subsections: "Black box" models; Breast Cancer; Hypothesis; Objectives; Thesis contributions and Sections of work.

1.1 "Black box" models

Artificial Neural Networks (ANNs) are good for prediction, but they are recognized to be "black box" models. This word refers to every system that can be observed in terms of its inputs and outputs, without knowledge of its internal workings (Zhang *et al.*, 2018). Any variable that enters any node in a network interact with many others through weights in the connections and activation functions. For this reason a change in any given input affect the ultimate result through several pathways. Therefore, the relationship between a given input and the output of the network is complex. For these interactions inside the network, it is hard to determine the contribution of one predictor variable among the rest.

To overcome this problem several methods to determine variable contribution have been tested and these can be classified two types. Methods where the inputs of the network change and techniques where the weights are handled to determine variable importance. These approaches are discussed in more detail in subsection 2.4.

Most of the studies to determine variable importance have been done in articles of ecology. For example, which inputs inhibit and which is the incidence on Anabaena spp, plankton growth (Maier 1998). These methods have been implemented in several articles of geotechnical engineering (Cao and Qiao 2008), chemical characteristics of honey (Péntos 2016), among many others.

A problem to figure out variable importance is that every time a neural network initialized the weights change. This affects the methods employed to determine relative importance and this is refereed as instability. To handle this problem, a set of ANN with the same architecture is used (Oña and Garrido 2014). In some

cases a neural network is initialized a 1,000 times to get reliable results (Hattab *et al.* 2013). The importance of studying black box models is they are significantly problematic to interpret. Another issue is that it is challenging to identify which variables, or descriptors are the most important and their relationship to the output property (Zhang *et al.* 2018).

Black-box methods have two key issues. These are determining (1) how the model outputs are derived, and (2) how the input variables affect the response (Lu *et al.* 2001). The second problem will be the focus of this dissertation.

1.1.1 Methods to determine variable importance and instability

To answer the question of the importance of each input in predicting the response variable, several methods have been used. Several of these methods based on a pure "black" box approach, this known as sensitivity analysis. The procedure consists of changing each one of the inputs and checking what happens to the output. Furthermore, there are alternative techniques based on the weights some of which consist of vector multiplications and others on the absolute value of the proportion of these parameters (Oña and Garrido 2014).

Some of the approaches used to rank the variable importance are partial derivative, second order partial derivative, profile, perturb, stepwise elimination, stepwise addition, stepwise improved a, stepwise improved b, holdback input randomization (HIPR), Monte Carlo sensitivity analysis, cluster base sensitivity analysis, data-based sensitivity analysis, one dimensional analysis, global sensitivity analysis, Garson, Connection Weights, among others. All of these methods are affected by instability. This means that with a different data set and implementing the same ANN architecture the final model achieved is different. Moreover, each time a neural network is initialized, the final model is different (Oña and Garrido 2014). A problem caused by this initialization is that in some cases the ANN do not converge. This also can be caused by the data set.

1.2 Breast Cancer

1.2.1 Definition and diagnostic methods

In this dissertation, to establish the variable importance a data set of Breast Cancer (BC) was handled and was taken from the Machine learning Repository.

Cancer is the abnormal growth of cells which tend to proliferate, and in some cases metastasize (spread to other organs in the body). These immortal cells develop tumors. There are many types of this disease, and they are termed after the organ where they grow. The most typical types of this disease comprise the following: prostate, breast, kidney, pancreatic, uterine, melanoma, lung, lymphoma, bladder and colorectal cancer (National Cancer Institute).

BC is the most prevalent cancer in women all over the world. It is one of the primary causes of death for women (Ragab *et al.* 2019) due to its prevalence and high mortality rate. Moreover, 80% of the cases of BC are invasive, this means the tumor cells can travel and grow in surrounding breast tissue. In 2017, there were approximately 252,710 cases of the present aggressive disease (American Cancer Society 2017).

Additionally, there are different diagnostic methods for BC. To understand how well these techniques work sensitivity, specificity and accuracy are needed to be defined. The first terms refers to the proportion of positive cases that were correctly predicted. The second term is the proportion of negatives that got predicted as negatives (Kumar 2018). The last definition refers to the ratio of correct predictions to the total number of observations (Mishra 2018).

The previous techniques can be classified in two types. In one hand, there are classical methods consist of breast examinations, mammogram, ultrasound, taking sample breast cells (biopsy) and magnetic resonance imaging (Mayo clinic 2019). The overall sensitivity and specificity of mammogram are 73% and in the case of ultrasound is 100% and 80.4% (Peprah and Sarkodie 2018).

On the other, artificial intelligence methods have been utilized to detect BC in the early stages. These include machine learning methods like Logistic Regression (LR), Random Forests (RF), Support Vector Machines (SVM). These techniques and other related approaches, which are neural networks and Extreme Boosting (EB), have been successful in identifying unknown variables. This is an advantage over software like Microsoft Excel, SPSS or STATA that are unable to identify unknown variables (Ganggayah *et al.* 2019).

In Patrício *et al.* (2018) a data set of 166 individuals with the variables Age,

BMI (Body Mass Index), Glucose, Insulin, HOMA (Homeostatic Model Assessment), Leptin, Adiponectin, Resistin and MCP-1 (Monocyte Chemoattractant Protein-1) were used to decide was healthy or has BC. To determine this, the above Machine Learning (ML) methods were applied. SVM have the specificity in the range of 85% to 90% and a sensitivity between 82% and 88%. The previous model used Glucose, Resistin, Age and BMI as predictor variables. A disadvantage of these methods is they are affected by the data set, location and lifestyle of patients (Ganggayah *et al.* 2019). Therefore, the results may not generalize to patients from other areas.

1.2.2 Artificial Neural Network used to predict BC

There are three types of ANN employed to predict BC, these are: Multilayer Perceptron (MLP), Convolutional Neural Network (CNN) and Deep Neural Network (DNN). The MLP consist on three types of layers (input, hidden and output). This network learns with labeled data. The CNN and DNN can learn from labeled, unlabeled and a mix of both type of data. The CNN employed the mathematical operation known as convolution for image detection (Schmidhuber 2015).

Multilayer perceptrons

An ANN was used by Lo *et al.* (1997). This neural network had one hidden layer and was used to predict BC. This ANN considered the characteristics collected from mammography. The model had seven input variables, eight hidden neurons, and one output. The predictor variables were mass margin, shape, density, calculate description, distribution, number, and patient age. The specificity of the model was 100% and the sensitivity was 71%.

Convolutional neural networks in BR

These networks have been utilized to asses BC risk. Images are employed to extract features classify as high or low risk. The model had been already trained in the classification of 1.2 million images into 1000 classes. The pre-trained network was used to analyze a region of interest of the input image of 227 $\times$ 227 pixels. The network contained 11 layers. Five of them were convolutional, three were fully connected, and the last three were pooling layers. The output vector of the first layer went to dimension reduction to eliminate the features with zero variance (Li *et al.* 2017). Another ANN was used to analyze bio-markers through images of mammography. This procedure consisted of implementing regression ANN for classifying malignant and benign tumors with an accuracy of 95.85 % (Rodriguez

et al. 2017).

Deep neural network in BR

An ANN with two hidden layers was used to predict breast cancer. To decide the optimum number of neurons in each layer, several types of architectures were implemented. Subsequently, to decide which one was more appropriate the error between the real and simulated value was determined. The optimum number of neurons in the first hidden layer was five and in the second hidden layer was three. This model included four inputs: tumor size, nodal status, age of patient and histological grade. This network employed the input variables to predicted the status of the estrogen receptor (Nastac *et al.* 2004).

Another ANN was used in Tariq (2018) to predict BC with a training, validation and test data sets. The overall sensitivity was 99.3%, a specificity of 100% and an accuracy of 99.4%. To determine if the tumor is benign or malign feature extraction was performed. To develop this procedure, a Gray Level Co-ocurrence Matrix was employed. This technique consists of extracting 10 characteristics. These features were contrast, sum of square of variance, entropy, sum average, sum entropy, sum average, difference variance, difference entropy, information of correlation and correlation. These characteristics were operated to determine whether the tumor was malign or not.

As we can observe different types of neural networks have been used for more than 20 years to improve the BC diagnostic. As mentioned in the previous paragraphs ANNs are good for BC prediction, but it is still hard to understand the relationship between the BC risk factors and this disease. This is due to the fact that it is hard to distinguish the effect that a particular variable has on a neural network. There are plenty of ANNs that are employed to capture nonlinearities in the data (Oña and Garrido 2014). The weights on these networks produce complex relationships between the inputs and the outputs. Put differently, all the inputs interact with each other in an intricate manner.

1.2.3 Variables involved in BC

To understand the importance of the predictor variables in an ANN, some of these variables are explained in this work. One of these variables is the age that as it increases the risk of developing BC increment. A woman in the USA from the age

of birth to 39 years has a probability of developing this disease of 1 out of 202. Then from 40 to 59 years the chances increment to 1 out of 26 and one out of 28 from 60 to 69 years (Shah *et al.* 2014).

Another critical factor involved in cancer development is the prior history of BC. People that have already developed BC has more probability of a second contralateral or ipsilateral BC (Shah *et al.* 2014).

It is equally essential to understand a feature that is the family history. This means that women with a mother diagnosed before 50 years old a relative risk of 1.69. In the case of women with a mother diagnosed, after the age of 50 years the risk is 1.37 (Shah *et al.* 2014).

Moreover, genetic predisposition is another critical predictor variable. 5% to 10% of BC cases have autosomal heritance. There are some crucial alleles that increase from 40% to 85% the risk of developing cancer. These are mutations in genes BRCA1 (Breast Cancer 1), BRCA2 (Breast Cancer 2), TP53 (Transform Protein 53), PTEN (Phosphatase and Tensin homolog), STK11 (Serine/Threonine Kinase 11), NF1 (Neurofibromin 1) and CDH-1 (Cadherin-1) (Shah *et al.* 2014).

1.2.4 Variable importance in BC

For discovering the relative variable importance in BC the following ML methods have been used. These methods are Generalized Linear Model (GLM), SVM, RF, GLM-Net, Partial Least Square (PLS), k-Nearest Neighbors (k-NN), ANN and Boosted Trees (BT). The previous techniques were employed in a comparative study to determine the rank of variable importance. This study established that the tumor size, cancer grade, number of positive lymph nodes and estrogen receptor are relevant variables (Boughorbel *et al.* 2016).

In addition in Ganggayah *et al.* (2019) some of the previously commented ML techniques were utilized. These ML methods comprised the following: Decision Tree (DT), RF, ANN, EB, LR, and SVM. In this case, the six key variables were cancer stage classification, tumor size, total axillary lymph nodes removed, positive lymph nodes, primary treatment type and method of diagnosis.

Additionally, in other work to determine variable importance other methods

were used. These were Cox-regression, RF; Random Survival 2 conservation-of-events, Random survival forest using log-rank, Random Survival forest using log-rank splitting and Random Survival Forest using log-rank score. From these methods, it was found that the highly predictive variables were positive nodes, age and progesterone. (Ishwaran *et al.* 2008).

Another specific example where they determined crucial variables is the following: a SVM model was employed to determine ten relevant predictor variables that contribute the most to BC survival. The most important variable was behavior of tumor and the least relevant was the stage of malignancy (Afshar *et al.* 2015).

1.3 Hypothesis

Sensitivity methods can be employed to establish variable importance in an ANN architecture. This will allow us to analyze the impact of each feature in the response variable. This relationship between the characteristics and BC is hard to observe. Three essential factors affect the contribution and rank of these variables: correlation, mislabeling probability and sample size.

1.4 Objectives

1.4.1 Main objective

The principal objective is to determine the ranking of importance of the nine variables (Age, BMI, Glucose, Insulin, Leptin, Resistin, Adiponectin, MCP.1 and HOMA in the case of real data and x1, x2, x3, x4, x5, x6, x7, x8 and x9 for simulated data) to the response. The techniques that will be used are the following: Connection Weights and Partial Derivative. Another critical aspect is to comprehend how the contribution and variable rank of these two techniques are affected by changing the correlation, mislabeling probability and sample size in simulated data. Subsequently, the obtained results are going to be compare to the same techniques applied to real information.

1.4.2 Specific objectives

1. Determining the ideal architecture using the relationship in the number of neurons and hidden layers through out the model performance metric MAPE

implementing the training and test data sets. This is going to be done in a collection of 50 ANNs.
2. Applying three variable importance methods (Connection Weights and Partial Derivative) to a set of 50 ANNs to find the contribution and rank of the real BC data set.
3. Simulating for the series of 50 ANNs three collections of data with 85, 510 and 1680 observation. In each cluster the correlation is going to be set to three levels: 0.1, 0.3 and 0.5 and the mislabeling probability is going to be set to 0.1, 0.2 and 0.3. In each case, the average absolute distance between the true and estimated variable ranks are calculated.
4. Implementing a program that creates an artificial neural network from scratch and compare its predicting power to the existing R functions.

1.5 Thesis contributions

Many of the articles comparing variable importance methods in ANN have being used in ecology modelling. Also, some of the articles used simulated data and most of the neural networks have only one hidden layer. A vast majority of the articles focus on finding the ideal architecture for an ANN with one hidden layer. In this work, the comparison between ANN with one hidden layer, two and three will be done. In this dissertation, real data will be employed to determine the best ANN architecture.

In most of the literature of variable importance the correlation among variables is low while in this work it is high. Another important contribution is to determine how well the variable ranking methods perform when some of the variables have a high degree of correlation or mislabelling and the sample size is small. The main contribution is to determine what is more important: the correlation, the mislabeling probability or sample size.

1.6 Summary

The present work is divided into five sections Theoretical Framework, Methodology, Results, Discussion, and Conclusions. In Theoretical Framework information about artificial neural networks, variable importance methods, correlation and mislabelling are given. In Methodology information about the data set chosen, the R implementation, the mathematics behind the chosen variable importance

methods and the steps in the methodology are shown. In Results, the boxplots and tables that show how ANNs and variable importance techniques perform are given. Additionally, their explanation is given in the same section. In the Discussion, an analysis of the results and a comparison with the literature is given. The information obtained in the thesis is given in the Conclusion section. There is an extra section that is the Appendix with useful results.

2 Theoretical framework

In this section a description of the features of artificial neural networks is given. Different types of optimization algorithms are shown. This part deals with how Multilayer Perceptrons (MLPs), which are employed to predict BC, work and their problems. Moreover, a description of the methods to determine the number of neurons and layers is given. Additionally, the variable importance approaches are shown.

This section is divided in four subsections: Artificial Neural Networks; Problems in training neural networks; Determining the ideal architecture; Description of methods to determine variable importance; Correlation and Classification with imperfect labels.

2.1 Artificial Neural Networks

The key feature that distinguishes ANNs from any other ML method is that they are assembled by nodes (neurons or processing units). These ML methods are classified in many types. Some of these ANNs are autoencoders used for pattern recognition (Chen *et al.* 2019); Hopfield networks applied to image detection and recognition (Wittek 2014); Convolutional Neural Network used in image recognition (Traoré *et al.* 2018); Long Short Term Memory that deals with times series prediction employed in Hua *et al.* (2018) and George *et al.* (2017); MLPs used in regression and classification tasks (Murtagh 1991); among others.

The chosen neural network type was MLP. A characteristic of this network is that in each hidden and sometimes in output neurons there is an activation function. This is what determines their degree of activity. Put differently, it is the function that determines if the neuron is switch off or on. This is the mathematical function that defines an artificial neuron. There are many types of activation functions some of these are: hyperbolic tangent, rectified linear unit, identity, softplus, gaussian, arctangent and sigmoid. Each of the previous functions produce different range of values (Sharma 2017).

The sigmoid or logistic was the chosen activation function because of its range of values. Its output range goes from 0 to 1 which is the same as the range of

values of the classification. Where 0 is patient and 1 is healthy.

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

Moreover, this network includes several layers of nodes with directional connections. The input of an activation function can be additive $input = \sum x_k w_k$. Another option is the multiplicative case $input = \prod x_k w_k$ (Schmidhuber 2015).

A MLP has artificial neurons, layers, weights, and connections. The connections represent the links between neurons that have one weight each one. The layers comprise a collection of nodes operating jointly in a specific depth. This type of network contains three types of layers: input, hidden and output as it can be observed in Figure 1. The artificial neuron is a mathematical function that received inputs and produces an output (Wikipedia 2019).

The weights or parameters comprise a collection of real-valued numbers. The. The weights in the input-hidden layer are the following set of vectors

$$(\alpha_{0m}, \alpha_m; m = 1, ...M) \qquad M(p+1) \tag{2}$$

The weights in the hidden-output layer are represented by the following collection of vectors

$$(\beta_{0m}, \beta_m; m = 1, ...K) \qquad K(M+1) \tag{3}$$

where p, M and K are the total number of neurons in the input, hidden and output layer, respectively. $M(p+1)$ and $K(M+1)$ are the total number of weights in input-hidden and hidden-output layers, respectively. The bias are error terms that are added to each neuron in the hidden and output layer to make the ANN model more flexible. The vectors α_{0m} and β_{0m} are bias added to the hidden and output neurons, respectively (Hastie *et al.* 2009).

This MLP possesses two relevant characteristics. It is a feedforward ANN that means its neurons can only be connected to processing elements of the following layer. All its neurons are completely connected (Kriesel 2005). The second characteristic is that it is a universal function approximator. Put differently, these systems can approximate any nonlinear function in a compact set. Additionally, is important to mention this type of ANN has been utilized in 70% of the litera-

ture of ANN (Oña and Garrido 2014).

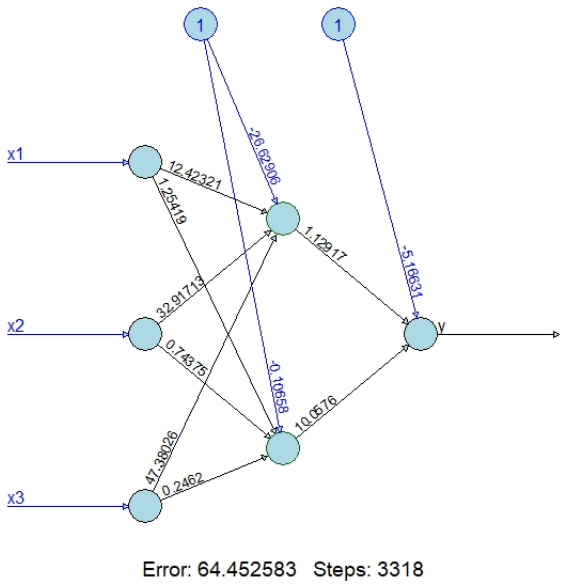


Figure 1: A MLP with three input, two hidden and one output neuron

Additionally, the characteristics of Z_k are formed by the linear combination of the inputs, and then the target Y_k is modeled as a function of the linear combinations of the Z_m. These equations are employed to describe the previous assumptions

$$Z_m = \sigma(\alpha_{0m} + \alpha_m^T X), \qquad m = 1, ..., M, \tag{4}$$

where X is the vector of the inputs. Z_m is a vector with a length M.

$$T_k = \beta_{0k} + \beta_k^T Z, \qquad k = 1, ..., K, \tag{5}$$

where T_k is the output vector of the hidden layer when the bias is added and has a length K.

$$f_k(X) = g_k(T), \qquad k = 1, ...K, \tag{6}$$

where $Z = (Z_1, Z_2, ..., Z_M)$ and $T = (T_1, T_2, ..., T_K)$ are output vectors of the input and hidden layers, respectively. The previous equations were taken from Hastie *et al.* (2009).

When there is only one output (Figure 1) the last two formulas are simplify to

$$T_1 = \beta_{01} + \beta_1^T Z, \qquad k = 1, \tag{7}$$

$$f_1(X) = g_1(T), \qquad k = 1, \tag{8}$$

where $Z = (Z_1, Z_2, ..., Z_M)$, and $T = T_1$. In this work and in the Figure 1 g_1 is the sigmoid function. This cannot be observed from the previous figure, but Neuralnet in R uses this function (Rdocumentation n.d.).

As we can see in the Figure 1 the first layer of neurons is where the input enters the MLP, then the input is multiplied by the weights in the connections and then added. This result is used to evaluate an activation function that determines the degree of activity of the neurons in the hidden layer. The error shows how well the ANN predict. The number of steps is the iterations before the neural network reach the final result. The weights shown in blue belong the bias and the ones connecting the neurons are shown in black.

2.1.1 ANN weight initialization

The process of determining the initial weights is refereed as initialization and there are several ways to initialize a MLP. Some of these methods are the following: all the parameters are set to zero (zero initialization); random method where the weights are distributed with a mean of zero and a standard deviation of one; the technique of Xavier in which the variance of a given node depends on the same value of the previous neuron; and He-et-al that is similar to the previous initialization with the difference that factor of Xavier is multiplied by two (Patel *et al.* 2018).

In the literature of variable importance the random approach was adopted. For the previous reason, all the ANNs employed in the dissertation used this type of initialization. Due to this type of initialization give you a different neural network model each time and this produces instability.

2.1.2 Types of learning and Batch size

The weights are used for the learning process. This means that after each iteration these parameters are updated. To adjust them, data is employed. There are three key types of learning that are the supervised that generates a model from

data with labels; unsupervised methods learn the common characteristics from class information without labels (Huang *at al.* 2018); and semi-supervised that combines a large amount of unlabeled information with labeled data (Zhu 2007). In this thesis, all the observations in the data were labeled. Therefore, supervised learning was used.

An epoch is the number of observations that passes through the network before updating the weights. Sometimes the entire data set is too large to being process and it is needed to be divided into batches. The batch size is the number of training examples in the batch. Iterations are the number of batches needed to complete one epoch.

2.1.3 Learning rate and Optimization methods

The first optimization algorithm successfully implemented in feedforward multilayer networks was backpropagation. This algorithm has two essential problems that are the convergence to a local minimum and the slow learning speed or rate. This last term is defined as the velocity at which the ANN learns from the data. The higher the learning rate the faster the optimization algorithm performs. This can lead to overshoot that is passing the minimum and not converging. In this case the final model obtained is not the optimum one. All types of learning require an optimization algorithm (Hastie *et al.* 2009).

To overcome these issues, several types of optimization methods have been implemented. Some of these approaches are modifications of the standard algorithm; techniques based on least-squares that minimize the MSE (Mean Square Error); second-order fast methods based on Quasi-Newton approaches like Levenberg-Marquardt and conjugate gradient algorithms and adaptive step size that is an adjustive acceleration strategy for convergence. In the second-order techniques, the most common updating procedures are Davidson-Fletcher-Powell (DFP) and Broyden-Fletcher-Goldfarb-Shanno (BFGS) (Castillo *et al.* 2006). In this work standard, RProp+ (Resilient Backpropagation with updating weights) and BFGS were used. The algorithms of these two methods are shown in sub-subsections 2.1.8 and 2.1.9.

2.1.4 Error functions and backpropagation

All of the previous optimization algorithms are iterative processes. To determine when to suspend the iterations a metric needs to be employed. This measurement is identified as the error function. Several error functions can be used. The Sum of Squared Errors (SEE) is used for neural networks that perform regression or classification:

$$R(\theta) = \sum_{k=1}^{K} \sum_{i=1}^{N} (y_{ik} - f_k(x_i))^2 \tag{9}$$

where the y_{ik} a vector of responses, $f_k(x_i)$ is a vector of predicted values, $R(\theta)$ that is the cumulative SEE from output node $k = 1$ to K and $i = 1$ to N observations (Hastie *et al.* 2009). Put differently, in some cases there are more than one output vector and the index k is employed to say which vector is using for the SEE according to the number of response variables.

In this dissertation their only one output. Therefore, k only takes the value of 1. The previous formula is simplified to

$$R(\theta) = \sum_{i=1}^{N} (y_{i1} - f_1(x_i))^2 \tag{10}$$

There are other types of error functions like the Cross-Entropy (CE). This is employed for classification. The formula of CE is the following:

$$R(\theta) = -\sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \log f_k(x_i) \tag{11}$$

To minimize $R(\theta)$ the weights need to change. To update these parameters in the rights direction backpropagation is employed. This consist on taking partial derivatives with respect to the weights β_{km} and α_{ml} of SEE is employed. Subsequently the chain rule is used and the following two formulas are obtained:

$$\frac{\partial R_i}{\partial \beta_{km}} = -2(y_{ik} - f_k(x_i)) g_k'(\beta_k^T z_i) z_{mi} \tag{12}$$

$$\frac{\partial R_i}{\partial \alpha_{ml}} = -\sum_{k=1}^{K} 2(y_{ik} - f_k(x_i)) g_k'(\beta_k^T z_i) \beta_{km} \sigma'(\alpha_m^T x_i) x_{il} \tag{13}$$

where g_k is an activation function, sometimes is the sigmoid function. x_i is the

input vector and $z_i = (z_{1i}, z_{2i}, ..., z_{Mi})$ is the output vector of the hidden layer. $z_{mi} = \sigma(\alpha_{0m} + \alpha_m^T x_i)$ is the output of a hidden layer neuron when an observation i pass through the network. x_{il} is the is one component of the vector x_i that correspond to the one input variable l (Hastie *et al.* 2009).

The first formula of the two is the chain rule applied with respect to a given weight β_{km}. The second equation is the output of taking the partial derivative with respect to a weight α_{ml}. This equation is larger because there are more mathematical operations between the input and output than between the hidden and output layers. The sum represents the case of multiple outputs, this means that a given input node can influence each output vector through all the connections between the input, hidden and output layers because the neurons in the network are fully connected.

In the case of this project there is only one output and as stated before is g is the sigmoid function. Then the formulas are the following:

$$\frac{\partial R_i}{\partial \beta_{1m}} = -2(y_{i1} - f_1(x_i))\sigma_1'(\beta_1^T z_i) z_{mi} \tag{14}$$

$$\frac{\partial R_i}{\partial \alpha_{ml}} = -2(y_{i1} - f_1(x_i))\sigma_1'(\beta_1^T z_i)\beta_{1m}\sigma'(\alpha_m^T x_i) x_{il} \tag{15}$$

2.1.5 Gradient Descent Backpropogation

This is an optimization technique that is used to find the minimum of any implicit function that relates the input vectors of the system to the output variable or vector; in the case of more than one output. In every iteration the weights are modified. This change in the $(r+1)$st iteration has the form:

$$\beta_{km}^{(r+1)} = \beta_{km}^{(r)} - \gamma_r \sum_{i=1}^{N} \frac{\partial R_i}{\partial \beta_{km}^{(r)}} \tag{16}$$

$$\alpha_{ml}^{(r+1)} = \alpha_{ml}^{(r)} - \gamma_r \sum_{i=1}^{N} \frac{\partial R_i}{\partial \alpha_{ml}^{(r)}} \tag{17}$$

where N is the number of training samples and γ_r is the learning rate and is a constant (Hastie *et al.* 2009). As it can be observed this two equations are similar to Newton-Raphson method due to the fact that they both used the first

derivative to find a minimum. The difference is that this method uses a partial derivative. It can be observed that the Jacobian matrix can be obtained from this approach.

If the learning rate is too small the optimization process is slow and takes many iteration to converge to a result. On the contrary, if the learning rate is too big the learning process is fast, but it can pass the minimum and not converge either. When a given threshold is met the iteration process stops.

2.1.6 Resilient Backpropagation (RPROP)

In this learning algorithm each weight is updated according to the sign of the partial derivative $\frac{\partial R}{\partial w_{ij}}$. w_{ij} represents both types of weights α_{ml} and β_{km}. Each weight is individually updated. This means that the step size of each weight is different. The step size is independent to the absolute value of the partial derivative.

An iteration of this algorithm consist of two parts. In the first part the step-sizes are adjusted. The step-size Δ_{ij} of a given weight w_{ij} is adjusted following these rules.
If $\frac{\partial R^{(t-1)}}{\partial w_{ij}} \frac{\partial R^{(t)}}{\partial w_{ij}} > 0$:

$$\Delta_{ij}^{t} := \eta^{+} \Delta_{ij}^{(t-1)} \tag{18}$$

If $\frac{\partial R^{(t-1)}}{\partial w_{ij}} \frac{\partial R^{(t)}}{\partial w_{ij}} < 0$:

$$\Delta_{ij}^{t} := \eta^{-} \Delta_{ij}^{(t-1)} \tag{19}$$

Otherwise:

$$\Delta_{ij}^{t} := \Delta_{ij}^{(t-1)} \tag{20}$$

where $0 < \eta^{-} < 1 < \eta^{+}$. If the partial derivative $\frac{\partial R^{(t-1)}}{\partial w_{ij}}$ has the same sign in consecutive steps the step size increases. On the opposite case if the sign changes from one step to the other the step-size decreases. The step-sizes are bounded by the following parameters Δ_{min} and Δ_{max}.

Resilient backpropagation with weights backtracking (RPROP+)
There are two cases. The first one is when the sign of the partial derivative remain unchanged, in this case the regular weight modification is performed:
If $\frac{\partial R^{(t-1)}}{\partial w_{ij}} * \frac{\partial R^{(t)}}{\partial w_{ij}} \geq 0$.

$$\Delta w_{ij}^{(t)} = -sign(\frac{\partial R^{(t)}}{\partial w_{ij}}) * \Delta_{ij}^{(t)} \tag{21}$$

if the partial derivative is positive the output of the sign operator is $+1$, if the partial derivative is negative the output is -1, and 0 otherwise. In the second case when the sign of the partial derivative is modified, the previous weight update is reverted:
If $\frac{\partial R^{(t-1)}}{\partial w_{ij}} * \frac{\partial R^{(t)}}{\partial w_{ij}} < 0$ then

$$\Delta w_{ij}^{(t)} := -\Delta_{ij}^{(t-1)} \qquad and \qquad \frac{\partial R^{(t)}}{\partial w_{ij}} := 0 \tag{22}$$

Fixing the derivative to zero prevents an update of the learning rate in the following iteration, because the "otherwise" option of equation (18) becomes active. The new weights are given by

$$w_{ij}^{(t+1)} := w_{ij}^{(t)} + \Delta w_{ij}^{(t)} \tag{23}$$

The above procedure was taken from Igel and Hüsken (2000).

2.1.7 Broyden, Fletcher, Goldfarb and Shanno (BFGS)

Quasi newton methods are used to solve unconstrained optimization problems. These methods used updating formulas for the approximation of the Hessian. BFGS was proposed in 1970 and has be widely used nowadays. The BFGS solves the following minimization problem:

$$min_\theta R(\theta) \tag{24}$$

where R is twice differentiable. This method uses f_i as an approximation of the Hessian G at x_i and q_i as an approximation of the gradient h. The BFGS formula uses the following step-vectors s_i and y_i

$$s_i = X_{i+1} - X_i \tag{25}$$

$$q_i = h(X_{i+1}) - h(X_i) = h_{i+1} - h_i \tag{26}$$

The following equation used to update R_i is the approximation of the Hessian G at X_i

$$R_{i+1} = R_i - \frac{R_i s_i s_i^T R_i}{s_i^T f_i s_i} + \frac{q_i q_i^T}{q_i^T s_i} \tag{27}$$

The previous equation satisfy the secant equation:

$$R_{i+1}s_i = q_i \tag{28}$$

It is only possible to satisfy the above equation if:

$$s_i^T q_i > 0 \tag{29}$$

The iterative formula for the quasi-Netwon methods is defined as

$$X_{i+1} = X_i + a_i d_i \tag{30}$$

where a_i is the step size and d_i is the search direction. The step size is always positive in order for $R(\theta)$ to be sufficiently reduced. The minimizing process depends on the choice of a_i and d_i. To perform this procedure the search line given by Armijo is used and it is the following:

Given $s > 0$, λ in the interval $(0, 1)$, τ in $(0, 1)$ and $\gamma_i = max(s, s\lambda, s\lambda^2, ...)$ such that:

$$R(x_i) - R(x_i + a_i d_i) \leq \tau \gamma_i h_i^T d_i \tag{31}$$

For $i = 0, 1, 2, 3, ...$ The reduction in f is proportional to the directional derivative $h_i^T d_i$ and a_i.

The direction of the quasi-Newton methods has the form of:

$$d_i = -R_i^{-1} h_i \tag{32}$$

where f_i is the symmetric and nonsingular matrix of approximation of the Hessian. The initial matrix f_o is chosen by an identity matrix which is updated by the formula of f_{i+1}. When d_1 is defined by the equation of d_i and f_i is a positive definite, we have $d_i^T = -h_i^T f_i^{-1} h_i < 0$, and for the previous equations d_i is a descent direction.

The above optimization method was taken from Ibrahim *et al.* (2014).

2.1.8 Fitting a neural network model

The RPROP+ and BFGS need data to minimize the function error. For these iterative processes sometimes three data sets are used. These are the training, validation and testing data sets.

The training data is employed to adjust a model to a neural network. This collection of observations is used to determine the values of the weights. The validation set is implemented to improve the previous parameters and achieve a more efficient performance in the training data. The third is employed to measure the performance of the neural network model without bias (Shah 2017). In this project, given the small sample size of the real data a training and test data set were employed.

2.2 Problems in training neural networks

2.2.1 Starting values

There are certain problems when data is employed to optimize an ANN. One of these is the initial values. If these values are nearby to zero the model starts out close to be linear and becomes nonlinear as the weights increases. If the parameters start in zero the ANN never learns. In the case when the weights are large, this leads to poor solutions. In this dissertation, the starting values are arbitrary numbers between -0.5 and 0.5 with a normal distribution (Hastie *et al.* 2009).

2.2.2 Overfitting

Another problem is overfitting. This means that the ANN predicts with high accuracy in the training data set, but poorly in the test data set. To avoid this problem regularization is employed. One approach used to regularize is weight decay. This method consists on adding a penalty term to the error function $R(\theta) + \lambda J(\theta)$, where

$$J(\theta) = \sum_{km} \beta_{km}^2 + \sum_{ml} \alpha_{ml}^2 \tag{33}$$

and λ is a constant.
There are other forms of penalty

$$J(\theta) = \sum_{km} \frac{\beta_{km}^2}{1+\beta_{km}^2} + \sum_{km} \frac{\alpha_{ml}^2}{1+\alpha_{ml}^2} \tag{34}$$

If the weights of the model are big then the error is also large. Therefore, as the error is minimized the absolute value of the weights decrease (Hastie *et al.* 2009).

In this project, overfitting was not addressed because there were only 116 observations in the real data. This sample size is too small to determine the 56 weights in the network. Therefore, the decision made was to focus only on finding the ideal architecture.

2.2.3 Number of hidden units and layers

Another problem is the number of hidden units and layers. These units manage to capture nonlinearities in the data. If there are too few of these, the network is unable to capture them. If the other case with too many, some weights can be decreased towards zero using regularization (Hastie *et al.* 2009). In this project, the number of neurons and hidden layers are changed. This will be performed to determine the optimum number of neurons and hidden layers.

2.2.4 Multiple minima

The last problem is that the target function has more than one minimum. Different initial values lead to different final models. Moreover including observations to a data set can produce an alternative model (Hastie *et al.* 2009). The point of having 50 repetitions of each type of network is to determine the average minimum.

2.3 Determining the ideal architecture

To select the ideal architecture, the performance of a model needs to be determined. For this procedure, several measurements can be employed used. Some of these are the Mean Error (ME), Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Standard Deviation of Errors (SDE), Mean Percent Error (MPE), Mean Absolute Percent Error (MAPE) (Masoud 2014), Mean Average Relative Error (MARE), R^2, r and d_2 (Shahabi *et al.* 2012).

From all of these measures the MAPE and MSE are employed in this work. The formula of MAPE is the following:

$$MAPE = \frac{1}{T}\sum_{i=1}^{T}\left|\frac{Real - Predicted}{Predicted}\right| \tag{35}$$

This equation was taken from Oña and Garrido (2014). MAPE was the chosen metric because is scale-independent and easy to interpret (Kim and Kim 2016). The other metric employed is the MSE given by

$$MSE = \frac{1}{T}\sum_{i=1}^{T}(Real - Predicted)^2 \tag{36}$$

The previous formula was taken from Shahabi *et al.* (2012). This metric is useful because we have unexpected values that we should care about (Drakos 2018).

To determine the number of hidden units, Oña and Garrido (2014) varied the number nodes in this layer from 1 to 30. To determine the ideal architecture, they calculated the value of MAPE for each network. To make the results reliable 50 ANN with different initial weights were trained. Subsequently, the average value and the standard deviation of MAPE was calculated.

In other works like Olden *et al.* (2004) the optimal number of neurons in the hidden layer was determined by cross-validation. And the perceptrons change from being 1 to 25. In Péntos (2016) the neurons in the hidden layer varied from 3 to 25.

Another approach employed to determine the ideal ANN architecture was done in Cao and Qiao (2008). In this method a neural network committee sensitivity analysis strategy was used and has the following steps:
1) Empirically choosing excellent ANNs as seeds for the neural network model from popular types of networks. These are good to handle nonlinearities due to the complexity in the data used to obtain the ANN models.
2) From every seed m ANNs are produced with diverse number of hidden layers and neurons in empirical ranges.
3) From each group of ANNs choose $k = 3/10m$ with the best performance.

4) From each explicative variables, calculate the mean of the corresponding rank.

2.4 Description of methods to determine variable importance

After selecting the ideal architecture the importance can be determined This is defined as how much an input affects the output. There are many types of error to determine the importance of a variable. Relative importance of variable one is given by the following formula:

$$R_{imp1} = \frac{abs(E_1)}{abs(E_1)..... + abs(E_n)} * 100 \tag{37}$$

R_{imp1} is the importance of one variable and is calculated with the previous formula. In some cases E is the magnitude of the weights and in others is the change in the MSE. The previous formula was taken from R/validann.R (n.d). In the following paragraphs the state of art of the variable importance method is given.

In Scardi and Harding (1999) a sensitivity analysis was used. This methodology was employed to discover the relationship between the response variable, which was the phytoplankton primary production, and the predictors. White noise was added to each feature, and the variations in the MSE was observed. This perturbation varied in the range of $[-0.1, 0.1]$ to $[-0.5, 0.5]$.

In contrast in Yao *et al.* (1996) an empiric approach recognized as the rule of thumb was used to establish variable importance in marketing data. The method consisted of omitting one attribute and run the ANN numerous times. If the results were the same before or better after removing the variable then it was inferred than this predictor was unimportant to the model. On the opposite case, provided the results deteriorate after the feature was eliminated subsequently this determines which variable was significant. The measure of R-square was employed to check if the model improves or deteriorates.

Moreover, in the same study, three other methods for sensitivity analysis were used. The first of these three techniques was weight sensitivity analysis. This approach employed the Equation method for a feedforward neural network with one hidden layer and one output. This equation included the values of the output

node, the hidden neurons and all the weights in the network. These values were used to determine the mean influence of each variable.

The second approach is the Weight Magnitude Analysis Method. For this approach, normalization was employed. This procedure consists of dividing the weights between the input and hidden nodes by the maximum value of all of these parameters. Next these normalized weights are summed.

The last approach was the Variable Perturbation Method. The perturbation consisted on $X_l = X_l + \delta$ or $X_l = X_l * \delta$ where X_l is a given variable. A predictor variable was perturbed while the others keep their original values. This method was improved in Kemp *et al.* (2007). In this other article, the values of each input variable were set to arbitrary numbers between 0.1 and 0.9. This is the range of the initial weights employed in the optimization process.

Another type of method known as the Profile Method has been employed in many papers. Some of these articles were two of Lek *et al.* (1996) and another of Lek *et al.* (1995). All of these three articles were about ecology modelling. The key feature of this approach is to generate a partial derivative of the response variable with respect to each descriptor. This allows for clarifying the complex relationships in the ANN model. It is also known as Lek´s profile method.

Moreover, in another article of ecology modelling of Olden and Jackson (2002) four methods were used to determine variable importance. These approaches were Garson, neural interpretation diagram, Lek´s randomization test and profile. The first three techniques are based on the values of the weights. The method of Garson is widely employed in the literature, and it is based on the absolute value of the weights to determine the relative importance. And in Olden *et al.* (2004) the method of Garson was found to be inaccurate. This is due to the fact that the absolute value of the weights avoid taking into account interactions that are in opposite directions and counterinteract. Another technique used in the previous work was the Weight Connection method based on a vector multiplication between the parameters on the input-hidden links and the ones in the hidden-output. This approach showed more accurate results that the rest of eight methods of variable importance applied on this work.

A less conventional method is the randomization test that consist on performing the Garson algorithm 1000 times with a randomly permuted response variable that is different each time. Then the statistical significance of each input was observed and three different distributions were obtained. These represent the distribution of the relative importance, overall connection weight and input-hidden-output connection weight. Also the p-value of the randomization test of each variable was obtained. Additionally, the method of the neural interpretation diagram where the relative importance of the parameters is represented by the thickness of the lines.

Another iterative algorithm was implemented in Hattab *et al.* (2013) Monte Carlo sensitivity analysis was carried out. This research was conducted to determine the variability of copper concentration in phytoremediated contaminated soils. This analysis consisted on generating a data set and each one of its variables have a random distribution. Subsequently the ANN prediction of copper is determined. This procedure was performed 1000 times and a statistical distribution function was acquired that relate each input to the response variable. An alternative method was employed in Lin and Cunningham (1995) that is called the fuzzy curves approach. The technique consisted on employing a exponential function to assign a group membership to each point. Then used an equation to defuzzify and obtain every part in the fuzzy curve.

In contrast an alternative approach was implemented for the first time in Dimopoulos *et al.* (1995).The technique is recognized as the partial derivative (Pad) method. This approach applies the Jacobian matrix to calculate the sensitivity of an output to a given change in an input. Another version of the Pad was proposed in Yeh and Cheng (2010) that considered the first and second derivative for sensitivity analysis. The procedure considered a linear and quadratic term produced by the first and second derivative, respectively. An approximation of the Pad method was proposed in Montaño and Palmer (2003) employing a numeric analysis. This methodology is based on the slope of each predictor variable and the output.

A different type of method was employed in Sung (1998).The method of stepwise for neural networks was proposed. This approach is based on the change in the MSE when an input is deleted. The rank of importance is determined by the

amount of change in this metric. The highest difference in the MSE determines which variable is more critical. In Gevrey *et al.* (2003) this method was used and it was termed as backward stepwise. To implement this method, they create all the nine models with everyone of the possible combinations of the eight predictor variables. Next they select the one with the highest MSE and produce the seven possible combinations omitting one attribute. This procedure was performed until all variables were gone. And the order in which they were eliminated is the rank of importance.

A contrasting method was implemented in Maier *et al.* (1998) that is forward stepwise. Their procedure consisted on generating ten models each one with one predictor variable. The one with the lowest MSE is the chosen one. Subsequently nine models were generated with the combination of the chosen variable and the rest of the predictors. This procedure was replicated until all the attributes were included. The order in which they were added is the rank of importance. Additionally, in Gevrey *et al.* (2004) two other approaches were adopted, two improved stepwise methods. Both of them use a single trained model. In the first case each variable is eliminated and the corresponding change in MSE is calculated. The higher the change in MSE the more important is the variable. This was named as improved stepwise a. In the second case all values of an input are set to its mean and the previous procedure is done again. This one is improved stepwise b.

2.5 Correlation

This term is a measurement of the degree of relationship between two variables. These can be positively correlated, as one increases the other increases and the opposite happens with a negative correlation. The following formula was taken from WallStreetMojo (n.d.)

$$r_{xy} = \frac{\sum(x_i - \bar{x})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}} \tag{38}$$

where x_i and y_i are the value of the predictor and response variable, respectively. $\bar{x}$ and $\bar{y}$ are the values of the mean of the predictor and response variable, respectively. There are two types of correlations. The one among the attributes and the one between each predictor variable and the response. There are few ar-

ticles where it is mentioned how correlation affects variable importance methods in ANNs. For this reason the following paragraphs information about other ML methods in the context of feature selection that is a related subject.

In Gregorutti *et al.* (2016) was mentioned that variable selection is hard when there is a correlation among the features. They used Recursive Feature Elimination to rank the variables. In this algorithm, the permutation importance measure is used. The previous procedure was performed in RF. The same algorithm was implemented in Sanz *et al.* (2018) by purposing three approaches to rank variables based on non-linear and survival analysis SVM.

Furthermore, in Paliwal and Kumar (2011) it was examined how the correlation affected two-variable importance methods. These methods were the Weights Connection and the modified Weights Connection based on the median interquartile range. They used using singular value decomposition to generate the correlated data. Additionally, in Szecówka *et al.* (2011) the Pad method was employed to determine which sensors were more redundant. Put differently, this indicates that when these sensors were removed the accuracy of the ANN decreased the least. In this procedure, an ANN with two hidden layers was used. Various combinations of two sensors were removed from the network. They obtained an architecture close to the optimal one.

In Hao *et al.* (2012) a variance-based importance measure was applied to ANNs. This approach is based on the fact that the contribution any attribute can be decomposed in the correlated and uncorrelated contribution with other input variables. Combining these two parameters an importance matrix of the contribution of correlated input variables was obtained. The previous approach was first purposed in Li *et al.* (2011). Mazurowski and Szecowka (2006) mentioned that the methods of sensitivity analysis are ineffective when the input variables are dependent on each other. In their study some of their variables represented a combination of others like $x_2 = \sin x_1$ or $x_2 = x_1^3$.

2.6 Classification with imperfect labels

Information about this subject in the context of variable importance was not found. Most of the reviewed literature was about how mislabelling affects ML methods and its accuracy. A related subject is how the performance of deep neu-

ral networks is affected by missclassication. A state of the art is showed in the following paragraphs.

Some data sets can have mislabelled observations. In these cases is important to understand how ML performs. Some of these methods are k-nearest neighbour (knn), SVM and linear discriminant analysis (LDA). It has been discovered that knn and SVM have a better performance than LDA when the sample size increases (Cannings *et al.* 2019). Brodley and Friedl (1999) mentioned that mislabeling occurred because of subjectivity, data-entry error or inadequacy of the information used to label. Their procedure demonstrated that filtering can improve classification accuracy up to 30%.

Moreover Patrini *et al.* (2017) mentioned how a specific ML method, which comprise a deep neural network, is affected by noise. They also said that two other sources of noise are the used of non-expert labellers and search engines because they assume that a keyword is a valid label. He proposed a robust approach to deal with mislabelling in deep neural networks. In Bekker and Goldberger (2016) a method that simultaneously learns both the neural network parameters and the noise distribution was used. This improved classification performance. In Martinez and Zeng (2001) an automatic data enhancement was used to constantly update a probability vector based on its difference from the output of the network. This vector was assigned to each training observation. As the iterations in the ANN occur, the mislabelling data become smaller.

Furthermore in Fard *et al.* (2017) it was demonstrated that non-uniform mislabeling is more challenging than the more frequently studied uniform case. Deep networks have inherent robustness when large datasets are available. Pechenizkiy *et al.* (2006) mentioned two types of noise that is wrong labeling class (mislabeling or misclassifications) and attribute (errors introduced to attribute values). The first kind of noise represents instances with the similar values of the predictor variables but different class labels, these are called irreducible or Bayes error. And the second type is wrongly classified observations that are known as mislabelings or misclassifications. In Rolnick *et al.* (2017) it was stated that neural networks can include noise-robust algorithms and also there are methods to discard or correct mislabeled data. Semi-supervised learning has been used to deal with mislabeling.

3 Methodology

Some of the methods stated in the previous section were employed in this one. The R code to produce the results is given in this part of the work. Moreover, information about the data set and the exploratory analysis is shown. In addition, this section includes how the optimization algorithms previously stated were implemented and the sample size, correlation the mislabeling probability were changed. Additionally, this part deals with employing the knowledge acquired in the Theoretical Framework.

This section is composed by four parts: data set description, determining the ideal architecture, variable importance methods and data simulation. In the first part the relationship among the variables is determine. In the second section the ideal number of neurons and hidden layer is found. In the third part a description of the variable importance procedures is given. In the last part the functions employed to simulated are described.

3.1 Data set description

To comprehend variable importance, a data set of Breast Cancer Coimbra was employed. This was taken from the Machine learning Repository and has 116 observations. The predictor variables in this set are the following: Age, BMI (kg/m^2), Glucose (mg/dL), Insulin (µU/mL), HOMA, Leptin (ng/mL), Adiponectin (µg/mL), Resistin (ng/mL) and MCP-1 (pg/dL). And the response variable is 2 for patients with BC and 1 for healthy people. The data is from a group of Portuguese women. This group is assembled by pre- and postmenopausal overweight women with and without breast cancer (Crisóstomo *et al.* 2016).

Lifestyle exerts a significant influence on the probability of developing breast cancer (Bruning *et al.* 1992) due to the fact that Insulin/Glucose homeostasis can be changed by habits. It was discovered that this equilibrium was altered in obese women (BMI $\leq 25kg/m^2$) (Luque *et al.* 2015). Obesity is related to the development of an aggressive BC because the adipose tissue is a microenvironment where the tumor cell can develop and progress to metastasis (Assiri *et al.* 2015). Sebastiani *et al.* (2016) mentioned that Age is also important. This is due to the fact that postmenopausal women with high BMI have higher probability of developing BC. Additionally, HOMA has been used to determine insulin resistance

that is a factor in BC development (Capasso *et al.* 2013).

Moreover, obesity has effects on a number of hormones like Leptin, Adiponectin, and Resistin. Leptin is one of the neurohormones that control the balance between the intake food and energy in the hypothalamus (Assiri *et al.* 2015). This hormone and its receptor are overproduced in some women with BC. This has been observed in 70%–80% of the cases (Artac and Altundag 2012). Resistin is a molecule that acquires its name for resistance to insulin. This hormone connects obesity to insulin resistance. High levels of this molecule are linked to breast cancer (Assiri *et al.* 2015). Adiponectin is a hormone that is a strong indicator of insulin sensitivity. Additionally, MCP-1 is an inflammatory chemokine related to the development and progression of BC. High levels of this molecule are linked to BC invasion and metastasis. This compound is more expressed in cancer cells than in any other type (Dutta *et al.* 2018).

3.2 Data treatment

The labels in the response variable changed from 2 that is a patient with BR to 0. The other class remain unchanged. Next the data was divided into two parts with 85 observations for training and 31 for testing. The observations in each set were taken randomly from the whole data.

After this procedure, the mean and standard deviation of each variable in the training data was employed to standardize both datasets. The boxplot of the data without standardization was produced. Another two boxplots were produced with the standardization of both types of data.

```
ind=sample(2,nrow(dat2), replace=TRUE, prob=c(0.7,0.3))
training=dat2[ind==1,1:10]
test=dat2[ind==2,1:10]
m=colMeans(training)
s=apply(training, 2,sd)
training2=scale(training, center=m,scale=s)
test2=scale(test, center=m,scale=s)
training2[,10]=training[,10]
test2[,10]=test[,10]
training=as.data.frame(training2)
```

```
colnames(training)=colnames(dat)
test=as.data.frame(test2)
colnames(test)=colnames(dat)
training$Classification=ifelse(training$Classification==1,1,0)
test$Classification=ifelse(test$Classification==1,1,0)
```

3.3 Correlation in the real data set

In most of the literature, the correlation is low (0.25 or lower) to obtain the variable contribution more easily. In this data set there is a high correlation among the predictor variables and this makes the variable importance analysis hard as mentioned in Section 2.5. Two solutions can be implemented. The first one is eliminating some of the variables until the correlation in the data is low enough. And the second approach is to employ methods of variable contribution to the data set with multicollinearity and determine their performance. In this work, the latter method was used.

3.4 Determination of ideal ANN architecture

The statistical programming language R was used to perform all the analysis in this work. The libraries used in R are neuralnet (Fritsch *et al.* 2019), NeuralNetTools, and Validann. The neuralnet function is used to create a neural network of the class nn and the function ann from the package Validann is used to create an object of class ann.

First of all, to determine the ideal ANN architecture, the function neuralnet was used. The data used to train the model was the standardized values. The threshold is the stopping criteria and is 0.01. The number of repetitions of the training process is one. The starting weights have a normal distribution. The algorithm was *rprop+* (Rdocumentation n.d.). The differentiable error function is the SEE. The activation function is logistic. The model fitted was the following:

$$\begin{aligned} X = w_{11}Age + w_{12}BMI + w_{13}Glucose + w_{14}Insulin + w_{15}HOMA \\ + w_{16}Leptin + w_{17}Adiponectin + w_{18}Resistin + w_{19}MCP.1 \end{aligned} \tag{39}$$

where w_{ij} is the weight between the hidden node j and the input neuron i. In

this case, the equation of X represents the input to the hidden node 1.

$$\sigma(X) = \frac{1}{1 + e^{-X}} \tag{40}$$

The previous equation determines how active is neuron 1. The same type of equation are employed in the rest of the nodes.

To determine the ideal architecture three types of neural networks were implemented. These were MLPs with one, two and three hidden layers. To create these types of networks. Two cycles *for* were used. The first one changing the number of layers and the second the number of neurons. Three variables were used to keep the number of neuron in each layer. There were three *if* for the three types of neural networks.

```
for (layers in 1:3) {
  for (neurons in 1:30){
    if(layerss==1){
      hidd_1[neurons] <- c(neurons)
      hidden_layers <- c(hidd_1)
    }
    if (layers==2){
      hidd_2 <- list(c(neurons, neurons))
      hidden_layers <- c(hidden_layers,hidd_2)
    }

    if (layers==3){
      hidd_3 <- list(c(neurons, neurons, neurons))
      hidden_layers <- c(hidden_layers,hidd_3)
    }
     print(hidden_layers)
    for (training in 1:length(hidden_layers)) {
      hidden = unlist(hidden_layers[training]
      ,use.names=FALSE)
    }
     for (r in 1:50){
     nn=neuralnet(Classification ~Insulin+Leptin+HOMA+MCP.1+
     Glucose+Adiponectin+Resistin+Age+BMI,
                  data=training,
```

```
                    linear.output = FALSE, hidden =hidden)
      prednn=predict(nn, training)
      f=as.matrix(training$Classification)
      a=mape(f, prednn)
      m[r, t]=a
      }
    t=t+1
  }
}
```

The previous R code was used with the training and test set. In the case of the test set this one was only used for predicting.

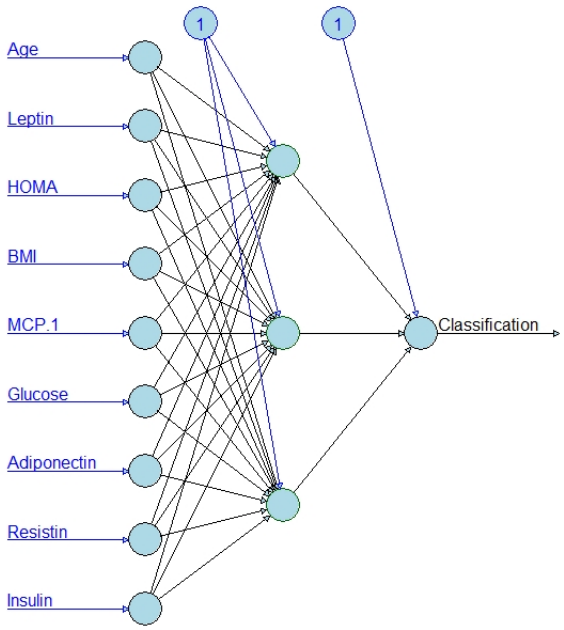


Figure 2: A MLP with one hidden layer

The first is a type of neural network with one hidden layer and changing the number of neurons from 1 to 30. To make this change, a variable was used to keep the number of neurons in the hidden layer.

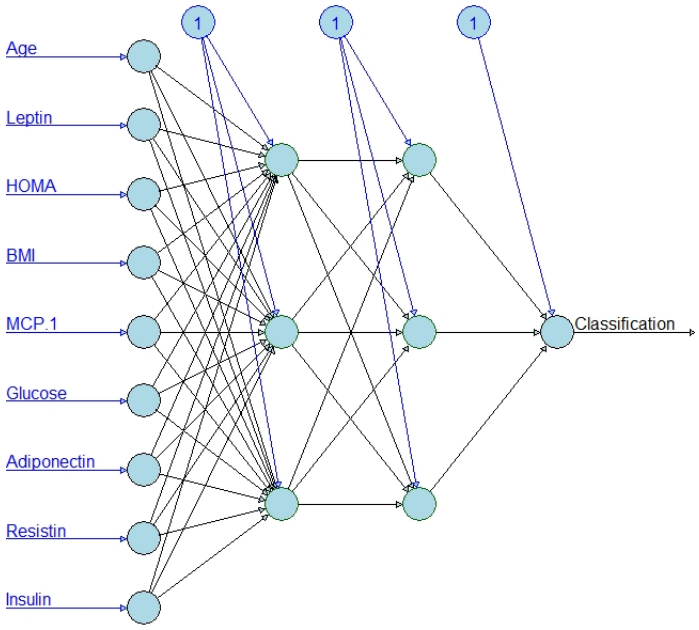


Figure 3: A MLP with two hidden layers

The second type is a neural network with two hidden layers and changing the number of neurons in both layers at the same time from 1 to 30. In this case, a vector of length two with the same number in both spaces was used.

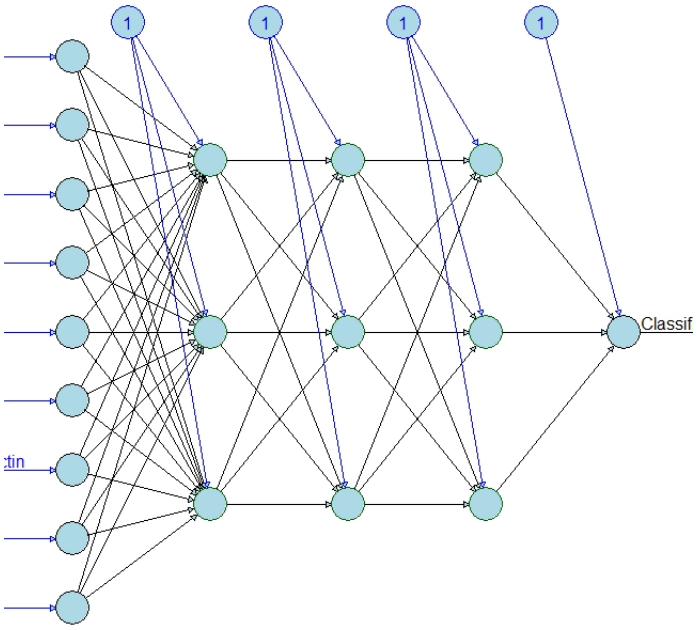


Figure 4: A MLP with three hidden layers

The last type is a neural network with three hidden layers and changing the number of neurons at the same time from 1 to 30. In this case, a vector of length three with the same number in all spaces was used.

In total, 90 different neural networks were implemented. To have reliable results, each network was initialized 50 times with different weights. This procedure is a modification from Oña and Garrido (2014). In this work they used only one hidden layer.

Statistical performance of the 50 ANN

For every network, the value of MAPE was calculated using the MAPE function in R. All of these values were kept in a matrix. From the column number 1 to column 30 the MAPE of the first type of ANN architecture with one hidden layer with the number of neurons increasing from 1 to 30. From columns 31 to 60 the values of this metric of the second type of architecture with two hidden layers. And columns 61 to 90 were used to store the values of this metric of the third type of architecture with three hidden layers.

A boxplot of the MAPE of 50 ANN with the same architecture was made.

This process was repeated for the 90 types of ANN. This was used to visualize the ideal MLP.

3.5 Determining variable importance

3.5.1 Ann

The ann function creates a neural network. The maximum number of iterations of this R function is 1000. Additionally, the optimization method employed is BFGS, the initial randoms weights are on the range of $[-0.5, 0.5]$, the number of hidden nodes is five and the activation function is sigmoid in the hidden and output layer (Denœux 2017).

3.5.2 Validann

This function performs goodness-of-fit, residuals analysis and computing structural validation metrics. Variable contribution is a result when performing the validation metrics. The output of this function is the relative importance of the nine variables with the following four methods: Garson, Weight Connection, Profile, and Partial Derivative (Rdocumentation n.d.). From these four techniques, the result of CW and Pad approaches were kept in row vectors each one with the relative importance determines by one of these two methods. Two matrices were created and each matrix was used to store 50 vectors of one type.

The Garson and Profile approaches were discarded. The first technique was eliminated because in Olden *et al.* (2004) it was found that it has the worst performance among nine variable importance methods employed (CW, Garson, Pad, Perturb, Profile, Forward stepwise, Backward stepwise, improved a and improved b). In a first attempt the Profile approach was implemented but its performance was worst than the other three methods.

The Pad and CW have been employed in several works. The first method was used in Yeh and Cheng (2010) and Dimopoulus *et al.* (1995). The other approach was employed in Paliwal and Kumar (2011) and Kemp *et al.* (2007). Both techniques were employed in Cai *et al.* (2015), Olden *et al.* (2002) and Mouton *et al.* (2010). For their wide used in the literature the two methods were implemented in this work.

For each method, a total of 50 ANNs with the same architecture were created. Every boxplot in the present work shows the result of initializing 50 times with different weights the same ANN.

```
for (i in 1:50){
  fit <- ann(x, y, size = 5, act_hid = "sigmoid",
  act_out = "sigmoid", rang = 0.5)
  obs <- y
  sim <- predict(fit, newdata = x)
  x=as.data.frame(x)
  result <- validann(fit, obs = obs, sim = sim, x = x)
  ress=as.matrix(result$ri)
  riCW[i,1:9]=ress[2,1:9]
  riPad[i,1:9]=ress[4,1:9]
}
```

3.5.3 Connection weights

This method uses the values of the weights to determine the importance of each predictor variable. It is a vector multiplication given by the following formula:

$$R_{lk} = \sum_{m=1}^{K} \alpha_{ml}\beta_{km} \tag{41}$$

This formula is only useful for neural networks with one hidden layer. R_{ij} represents the relative importance of the variable X_l with respect the predictor variable in the output neuron k.

3.5.4 Partial derivative

This method is referred to as the change of the predicted response variable with respect to a given predictor variable.

$$Z_m = \frac{1}{1 + e^{-net_m}} \tag{42}$$

This is the output of a neuron m in the hidden layer

$$net_m = \sum_{l} \alpha_{ml} X_l - \alpha_{0m} \tag{43}$$

$$f_k = \frac{1}{1 + e^{-net_k}} \tag{44}$$

where f_k is the output of the output layer of a neuron k.

$$net_k = \sum_k \beta_{km} X_k - \beta_{0m} \tag{45}$$

net_j and net_k represents the sum of all the weights in the connection in the hidden and output neurons, respectively.

$$\frac{df_l}{dx_l} = \sum_k \frac{df_k}{dnet_k} \frac{dnet_j}{dZ_k} \frac{dZ_k}{dnet_k} \frac{dnet_k}{dX_l} \tag{46}$$

$$= \sum_k f'_k \beta_{km} f'_k \alpha_{ml} \tag{47}$$

This equation represent the change in an output value for a given change in an input variable. To obtain this change the chain rule was applied.

$$f'_k = f_k(1 - f_k) \tag{48}$$

$$f'_k = Z_k(1 - Z_k) \tag{49}$$

When the derivative is applied to the sigmoid function the result is $e^{-x}/(1+e^{-x})^2$ where x is net_k or net_k. This result can be express as the two previous equations. The derivative of net_k with respect to Z_k is only the weight β_{km} because the remaining terms in the sum do not depend on this weight. The sensitivity for each variable is given by:

$$L_l = \frac{\sum_n \frac{df_k}{dX_l}}{n} \tag{50}$$

where n is the number of observations in the data set. The previous procedure was taken from Oña and Garrido (2014).

3.6 Data simulation

To determine how the sample size, mislabelling and correlation affect the variable importance contribution and rank. Data was simulated with three levels for each feature. Two R functions were used to create data rnorm and mvrnorm. This two functions are going to be explained in the sections Sample size and Correlation.

3.6.1 Sample size

Several studies in the variable contribution on ANNs used different sample sizes. In Oña and Garrido (2014) the sample size was 858 surveys. In Olden *et al.* (2004) the sample size was 10,000 simulated observations. In Cao and Qiao (2008) the sample size was 168.

Moreover, in Paliwal and Kumar (2011) three sample size were used: 60, 510 and 1680. These are small, medium and large, respectively. For this reason in the present work it was decided to used different sample sizes. The levels in sample size were 85, 510 and 1680. The function rnorm was used to created nine numeric objects that kept the values of the nine variables. These numbers were normally distributed and their mean was zero. The probability that $y = 1$ where y is a Bernoulli random variable is:

$$P = \frac{1}{1 + e^{-z}} \tag{51}$$

where $z = x * \beta$. x is a vector with the nine variables and β with the coefficients of the nine variables $(9, 8, 7, 6, 5, 4, 3, 2, 1)$. The last vector indicates the rank of importance in descending order. The output of the function P is zero or one.

```
x1 <- rnorm(n)
x2 <- rnorm(n)
x3 <- rnorm(n)
x4 <- rnorm(n)
x5 <- rnorm(n)
x6 <- rnorm(n)
x7 <- rnorm(n)
x8 <- rnorm(n)
x9 <- rnorm(n)
beta <- c(9,8,7,6,5,4,3,2,1)
z <- cbind(x1,x2,x3,x4,x5,x6,x7,x8,x9)%*%beta
prob <- 1/(1+exp(-z))
y <- rbinom(n, size=1, prob=prob)
```

3.6.2 Mislabeling

In this case the sample size changed from 85, 510 and 1680. The probability of corruption had three levels 0.1, 0.2 and 0.3. The function "which" was employed

to corrupt some of the outputs of the vector with the given probabilities. A total of nine ranks were produced. This procedure was performed to determine how the mislabeling probability affects the variable contribution and rank.

```
corruptprob<-0.1
whichcorrupt<-which(rbinom(n, size=1, prob=corruptprob)==1)
ycorrupt<-y
ycorrupt[whichcorrupt]<-1-y[whichcorrupt]
```

3.6.3 Correlation

The sample size changed from 85, 510 and 1680. The function mvrnorm was used to create a data set with a given correlation matrix. This means that a total of nine ranks were produced. This procedure was performed to determine how the correlation affects the variable contribution and rank. As the sample size increases the matrix Sigma is more similar to the output correlation.

```
Sigma <- matrix(.1, nrow=9, ncol=9) + diag(9)*.9
mu <- rep(0,9)
rawvars <- mvrnorm(n=n, mu=mu, Sigma=Sigma)
```

3.7 Methods of variable importance applied to simulated data

After obtaining each simulated data set the two methods of variable importance were implemented. This procedure was performed to determine if the small sample size, mislabeling and correlation can cause this two techniques to fail in real data sets. The same code for determining the variable importance in real data was employed for simulated data.

3.8 Stepwise improved a

This procedure consisted on training the model. Moreover, the training set was also used for prediction. Subsequently, each one of the inputs was omitted from the observations and used to predict an output vector. Then the change in the MSE was calculated employing the output vectors with all the variables and when each one was eliminated from the data set (Olden *et al.* 2004). This procedure was repeated with 50 ANNs using different initial weights.

```
for (e in 1:9){
  for (r in 1:50){
  nn=neuralnet(Classification ~Insulin+
               Leptin+HOMA+MCP.1+Glucose
               +Adiponectin+Resistin+BMI+Age,
                    data=training,
                    linear.output = FALSE, hidden =5)
  pr.nn <- predict(nn,training)
  MSE.nn1 <- sum((training$Classification - pr.nn)^2)/
  nrow(training)
  trainingm[1:85,i]=0
  x=trainingm
  nn=neuralnet(Classification~Insulin+Leptin+HOMA
              +MCP.1+Glucose+Adiponectin+Resistin
               +BMI+Age, data=trainingm,
               linear.output = FALSE, hidden =5)
  pr.nn <- predict(nn,trainingm)
  trainingm =as.data.frame(trainingm)
  MSE.nn2 <- sum((trainingm$Classification - pr.nn)^2)/
  nrow(training)
  deltaMSE=MSE.nn1-MSE.nn2
  m[t,i]=deltaMSE
  t=t+1
  trainingm=trainingk
  }
  t=1
  i=i+1
}
```

3.9 Building an ANN from scratch and comparing its performance to Ann and Neuralnet

This code that produces an ANN was taken from Tychobra (2018). In this work a explanation of this code is given.

The matrix of observations is given by

```
X <- matrix(c(
  0,1,0,
  0,0,1,
  1,1,0,
  1,0,1
),
  ncol = 3,
  byrow = TRUE
)
```

The observed outcomes are

```
y <- c(1, 1, 0, 1)
```

A vector with the initial weights between the input and hidden layer is created

```
rand_vector <- runif(ncol(X) * nrow(X))
```

This vector is employed to create a matrix

```
rand_matrix <- matrix(rand_vector,nrow =ncol(X),
ncol = nrow(X),byrow = TRUE)
```

The list that keeps the elements in the ANN is created.

```
my_nn <- list(
  input = X,
  weights1 = rand_matrix,
  weights2 = matrix(runif(4), ncol = 1),
  y = y,
  output = matrix(rep(0, times = 4),ncol = 1)
)
```

The weights2 are the parameters in the connections of the hidden to the output layer and output is the simulated response.

The activation function and its derivative is the following

```
sigmoid <- function(x) {
  1.0 / (1.0 + exp(-x))
}
sigmoid_derivative <- function(x) {
  x * (1.0 - x)
}
```

The Sum of Square Error is given by

```
loss_function <- function(nn) {
  sum((nn$y - nn$output) ^ 2)
}
```

The feedforward process consist of evaluating the sigmoid function using a vector produced by the matrix-vector multiplication and kept in the variable layer1. The previous function is also evaluated using the outcome of the matrix-vector multiplication to produce to overwrite the variable output.

```
feedforward <- function(nn) {
  nn$layer1 <- sigmoid(nn$input %*% nn$weights1)
  nn$output <- sigmoid(nn$layer1 %*% nn$weights2)
  nn
}
```

The backprogation algorithm is given by

```
backprop <- function(nn) {
  d_weights2 <- (t(nn$layer1) %*%
    (2 * (nn$y - nn$output) *
    sigmoid_derivative(nn$output))
  )
  d_weights1 <- ( 2 * (nn$y - nn$output) * sigmoid_derivative
  (nn$output)) %*% t(nn$weights2)
  d_weights1 <- d_weights1 *sigmoid_derivative(nn$layer1)
  d_weights1 <- t(nn$input) %*% d_weights1
  nn$weights1 <- nn$weights1 + d_weights1
  nn$weights2 <- nn$weights2 + d_weights2
  nn
}
```

*dweights*2 represent each one of the partial derivatives with respect to a given weight in the hidden-output connections. *dweights*1 is each one of the partial derivatives with respect to a given weights in the input-hidden connections. Then the parameters in both layers are updated adding these two results.

The training of the model is given by

```
n <- 1500
```

```
loss_df <- data.frame(
  iteration = 1:n,
  loss = vector("numeric", length = n)
)
for (i in seq_len(1500)) {
  my_nn <- feedforward(my_nn)
  my_nn <- backprop(my_nn)
  loss_df$loss[i] <- loss_function(my_nn)
}
data.frame(
  "Predicted" = round(my_nn$output, 3),
  "Actual" = y
)
```

n is the number of iterations. In each iteration the vector *lossdf* is employed to keep the values of the loss function. The final approximation of the output is represent by the vector Predicted. Then the metric MAPE was used to compare the performance of 50 ANNs created with the previous algorithm and the R functions Ann and Neuralnet.

4 Results and discussion

The first subsection is about exploratory analysis performed to the real observations. Moreover the methods of variable importance are employed in the training data set. Then to understand the results shown simulated data was employed. Subsequently the identical procedures to determine the contribution and variable rank was used as previously.

4.1 Real data analysis

4.1.1 Exploratory analysis

For this work, this will be the abbreviations Age (Ag), BMI (B), HOMA (H), Glucose (G), Insulin (I), Resistin (R), MCP.1 (M), Adiponectin (Ad) and Leptin (L). The methods were abbreviated like the Connection Weight method as CW and Partial derivative method as Pad. The following table is about the relationship among the predictor variables.

	A	B	G	I	H	L	Ad	R	M
A	1.000	-0.020	0.152	-0.020	0.020	-0.083	-0.169	-0.098	-0.064
B	-0.020	1.000	0.162	0.266	0.251	0.658	-0.321	0.133	0.225
G	0.152	0.162	1.000	0.346	0.498	0.173	-0.087	0.012	-0.077
I	-0.020	0.266	0.346	1.000	0.975	0.345	-0.024	0.110	0.028
H	0.020	0.251	0.498	0.975	1.000	0.330	-0.029	0.090	0.001
L	-0.083	0.658	0.173	0.345	0.330	1.000	-0.115	0.138	-0.003
Ad	-0.169	-0.321	-0.087	-0.024	-0.029	-0.115	1.000	-0.244	-0.103
R	-0.098	0.133	0.012	0.110	0.090	0.138	-0.244	1.000	0.321
M	-0.064	0.225	-0.077	0.028	0.001	-0.003	-0.103	0.321	1.000

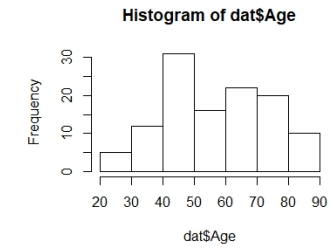
Table 1: Correlation in data

In the data set the G, R and H have the highest correlation with the response variable. While the variables A, L, and M are the least correlated with the response. In the case of the correlation among the predictor variables. The variables more correlated are H, I, G and B. There low correlations are between A and B, also between R and A. B and L have a moderate correlation. Many of the correlations are higher than 0.25 that is in Péntos (2016). And the relationship between the predictor variables and the response variables can be observed in the following table.

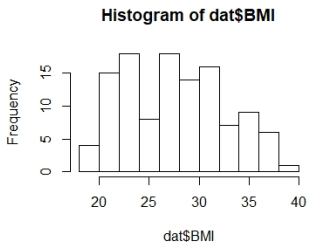
	A	B	G	I	H	L	Ad	R	M
y	0.043	0.167	-0.4	-0.179	-0.227	0.007	-0.075	-0.293	0.026

Table 2: Correlation with response variable

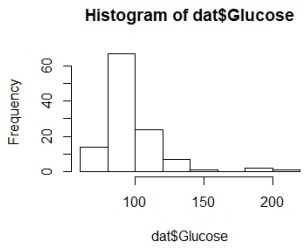
The three most important variables are G, H, and R and the three least important are A, M, and Ad. In the correlation of the data, we can see three high values in three variables (I, H and G) and one high value between B and L.



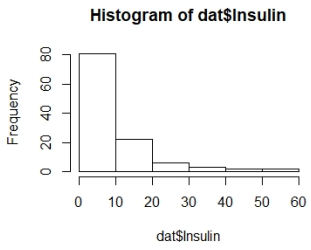
(a)



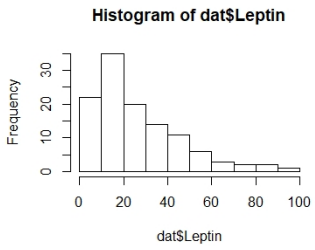
(b)



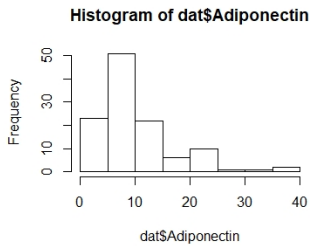
(c)



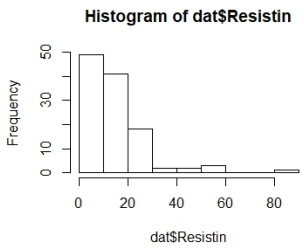
(d)



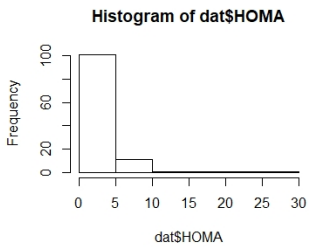
(e)



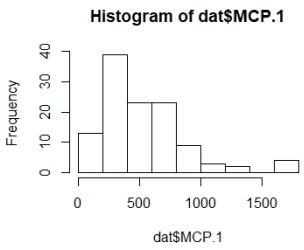
(f)



(g)



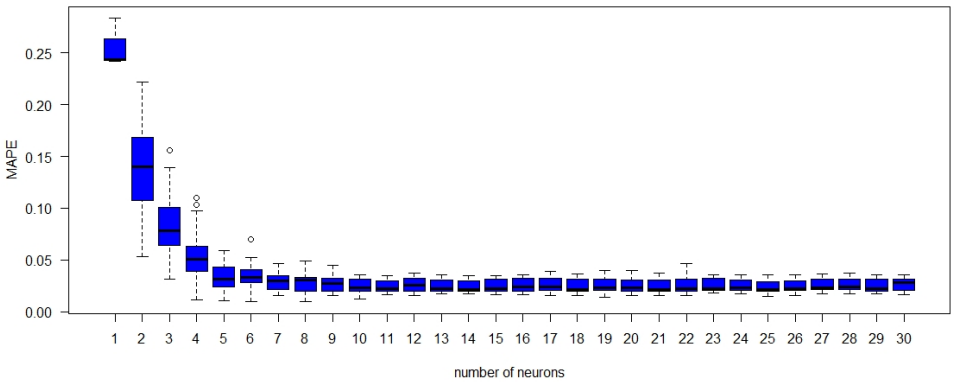
(h)



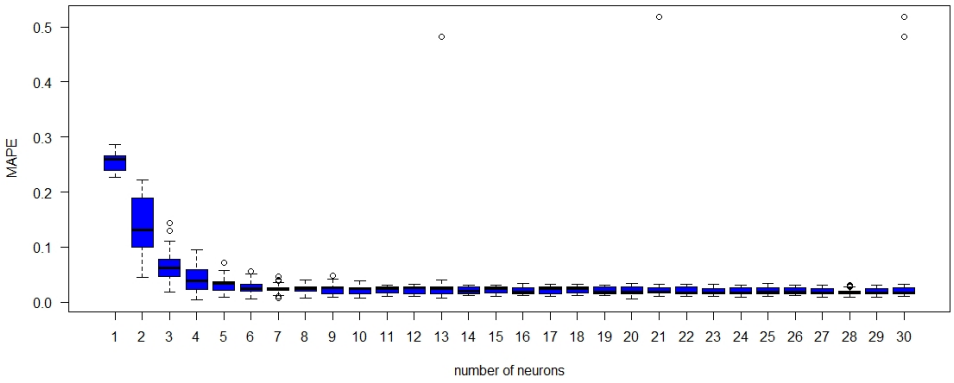
(i)

The mean, variance and distribution of each variable is different from the others. The ANNs are useful predicting when the predictor variables have different distributions. In the following figures another important finding is given. It is shown how the MAPE changes as the number of neurons and layers increases. This was performed with the training and test sets.

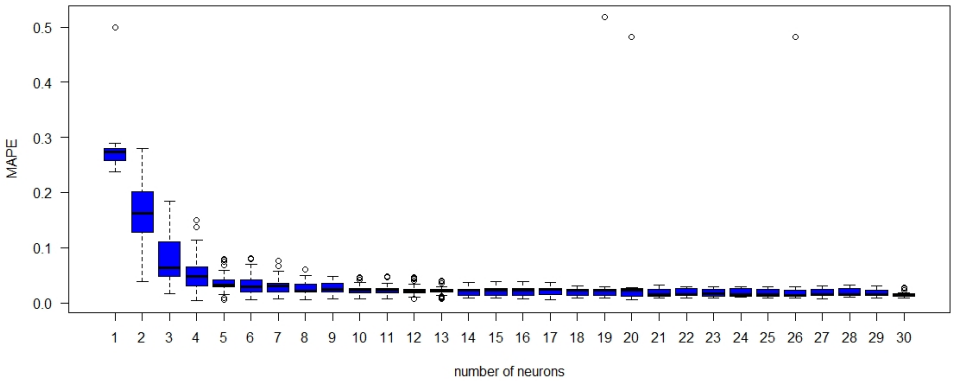
4.1.2 Determining ideal ANN architecture



(j) One hidden layer

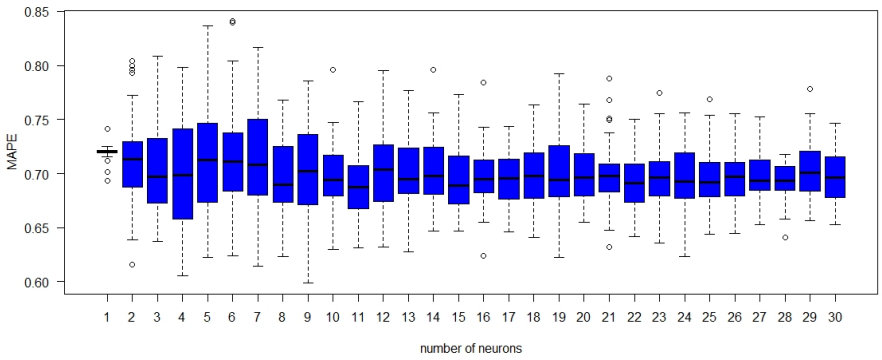


(k) Two hidden layers

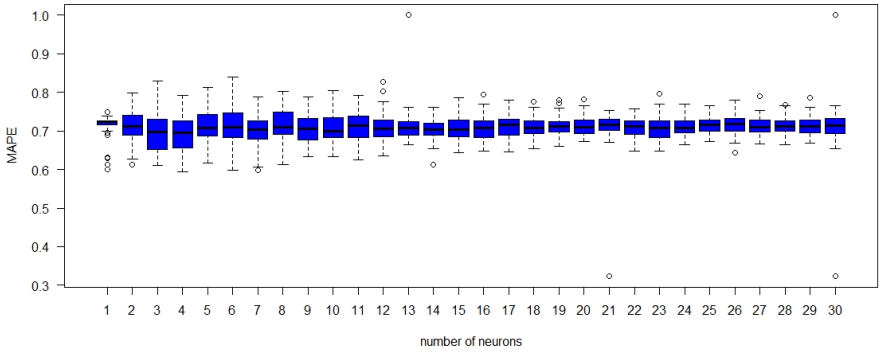


(l) Three hidden layers

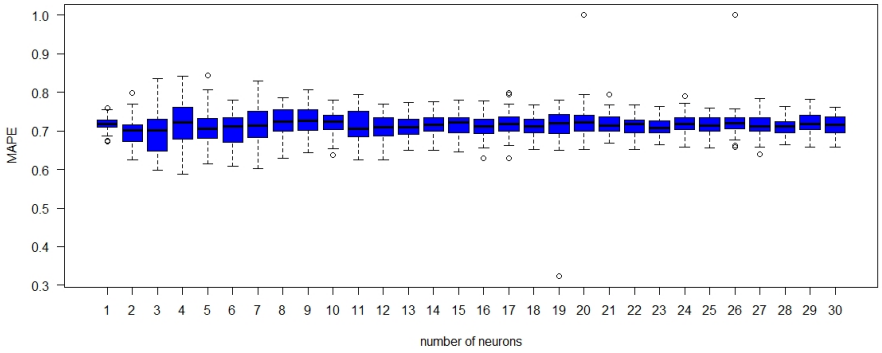
Figure 5: The behaviour of the function MAPE as the number of neurons increases until 30 with training set



(a) One hidden layer



(b) Two hidden layers



(c) Three hidden layers

Figure 6: The behaviour of the function MAPE as the number of neurons increases until 30 using the test data set

In the Figure 6, the distance from the minimum to the maximum increases as number of processing units changes from one to six. The median fluctuate and the number of neurons increases. After five neurons the size of interquartile range variates. The biggest interquartile range is achieved with two neurons. In the three cases the function MAPE decreases exponentially as the number of neurons increases. The position of the outliers change when more hidden layers are added.

In the Figure 7, the interquartile changes as the number of neurons increases. When the hidden layers increase to two and three the variability of MAPE decreases. The outliers change as the hidden layers increases. The training set has lower values of MAPE than the test set for this reason the only data employed for the following procedures is the training set. From the results it is deduced that increasing the number of hidden layers does not improve MAPE.

The function measuring the error performs worst in the test observations than with the training data. This showed that overfitting occurred. The model was trained with 85 observations in each iteration. To improve the models the set of observations must be divided in batches and pass through the network. In this work, overfitting is unconsidered as a significant problem because the number of observations is reduced. Additionally, as the sample size is small dealing with this problem would not benefit improving the accuracy of the variable importance methods.

A measurement was employed to determine the ideal ANN architecture. The chosen metric was MAPE because Oña and Garrido (2014) employed it to determine the performance of a collection of 50 ANNs. This measurement had higher values with the test than with the training set. This means the ANN was unable to generalize well the features in the training observations and these model have overfitting. Moreover, using the test observations as the number of hidden layer increases to two and three the variability decreases.

MAPE decreases as the number of neurons increases when the training set is employed. When this number increased to two and three the MAPE did not significantly improve. Therefore, accuracy did not raise as this value grows. The value of the outliers increment as the number of hidden units increased. The value of MAPE indicates that more than 95% of the people were classified correctly.

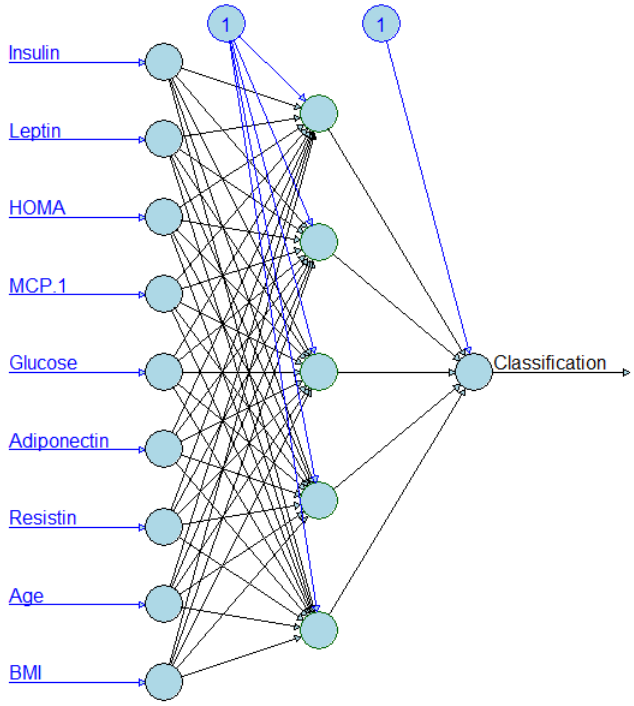


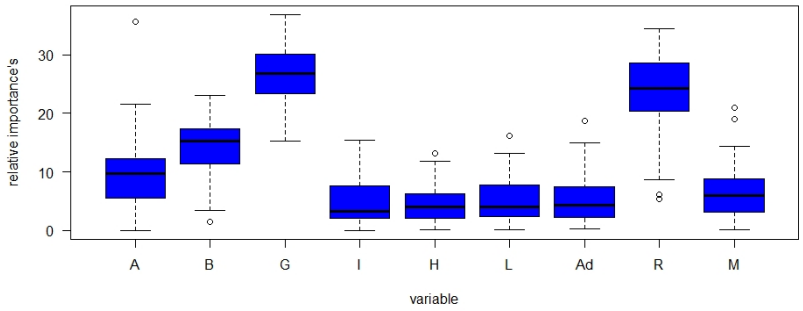
Figure 7: A MLP with nine inputs, five hidden and one output neuron

The previous analysis showed that the best architecture is a MLP shown in Figure 8. The input layer contained nine neurons, each one corresponding to one of the predictor variables. Then this layer is fully connected to five hidden units with a bias each one. These neurons and a bias are connected to an output unit that gives the classification of a person.

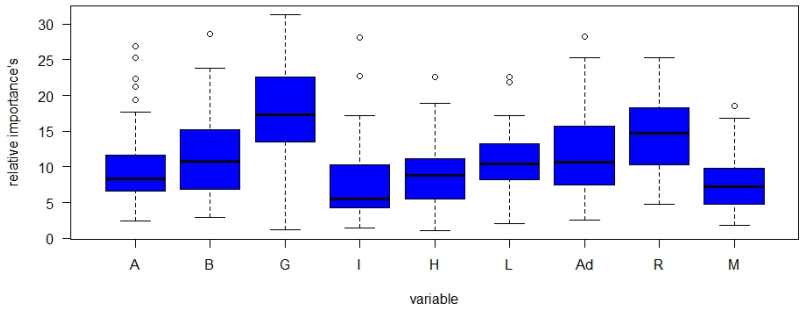
An important finding from the previous analysis is the ideal architecture has five hidden neurons. This is similar to Oña and Garrido (2014) where it was found that the optimum architecture was six neurons. They used 10 input variables and in this case nine predictor variables were employed. In both works, there was only one output variable.

4.1.3 Methods of variable importance

This ANN architecture was used to employ the methods of variable importance. This are shown in the following figures the CW and Pad approaches.



(a) CW



(b) Pad

Figure 8: Relative importance of the variables using both methods

The interquartile range of the boxplots of both methods is different. The Pad techniques has a bigger interquartile range than CW. In both cases there are outliers. The variables A, B, G, R and M are similar in both approaches. In CW, I, H, L and Ad have a similar relative importance while in Pad they have increasing importance.

The mean of the vectors used to produce the boxplots is the Average relative importance (Av). The Rank (Ra) is the descending order of the Av values where 1 represent the most important variable and has the highest Av and 9 is the least important and has the lowest Av.

Derive importance with real data				
	CW		Pad	
V	Av	Ra	Av	Ra
A	9.94	4	10.09	6
B	14.26	3	11.45	4
G	26.22	1	17.39	1
I	4.84	8	7.41	9
H	4.64	9	8.94	7
L	5.09	7	10.60	5
Ad	5.28	6	11.55	3
R	23.27	2	14.82	2
M	6.46	5	7.75	8

Table 3: Derive importance with real data

The only variables in the same order are G and R. The AV contribution in both methods is different. The biggest and lowest variable contributions were found with the CW approach.

Both methods produced different Av and Ra. The real rank of importance is unknown. Therefore, it is not possible to determine which method is more accurate. The Pad approach had bigger variability and more outliers than CW.

The methods of variable importance were employed to real data. The results changed from one approach to the other. The average importance is unique in each method. In some cases, this value was similar between approaches, but the rank was different. The minimum relative importance was 0.367 that corresponded to L. the maximum value of R was 26.215. To determine why these two methods were different. The correlation matrix was calculated. It was detected that there is high relationship among some variables in the data set.

Then to find if the correlation was the reason for this change several data sets with different levels of correlation and same sample size were simulated. This showed that even with small values of correlation, the variable contribution was affected when the sample size was small. Therefore, this is one of the reasons the techniques did not work properly in the real data.

The different ranks of importance can be due to several reasons. One of this reasons is that there is a high correlation between I and H and also among G and H. The information in the dataset is redundant and several variables are related to each other. In Pentós (2016) the maximum correlation between the predictor variables was -0.25. In this work, the maximum value of the previous factor is 0.932 that is three times more than the one in the previous article.

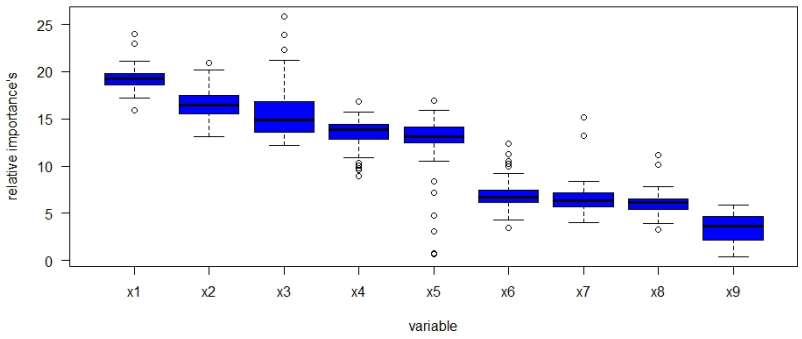
Additionally, all the method are affected by instability. This produces different distributions of variable importance and outliers. As in other works the fact that many values are required to initialize a neural network makes the rank of importance hard to determine. And different optimization algorithms could lead to unique models.

In Olden *et al.* (2004) the CW approach was the most accurate for ranking the importance of the variables. On this work, it is demonstrated that both approaches have similar performances. The rank of importance is different from the technique of G. Even though in both methods G and R are among the three more key variables. This demonstrated that Oña and Garrido (2014) and Pentós (2016) were right about that the Pad approach has a higher variability than CW.

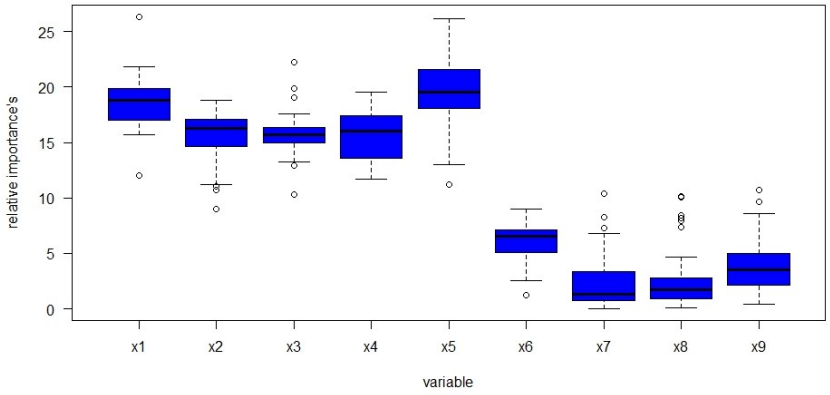
Both methods show almost the same findings. Therefore, the boxplots of only one approach are shown in the results, and the other one is in the appendix. The boxplots when changing the sample size, correlation and the mislabeling probability are similar for both approaches.

4.2 Simulated data analysis

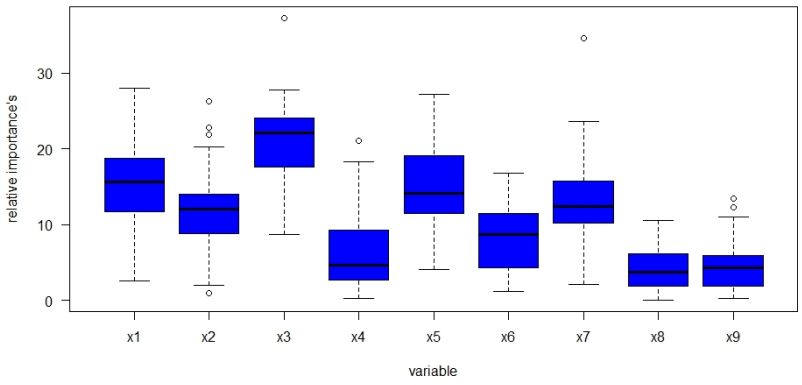
4.2.1 Methods of variable importance



(a) Relative importance



(b) Relative importance when correlation is 0.3



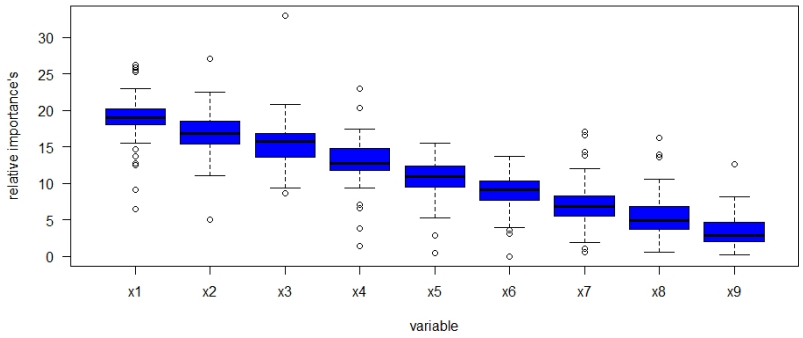
(c) Relative importance when the mislabeling probability is 0.3

Figure 9: CW method with sample size 85

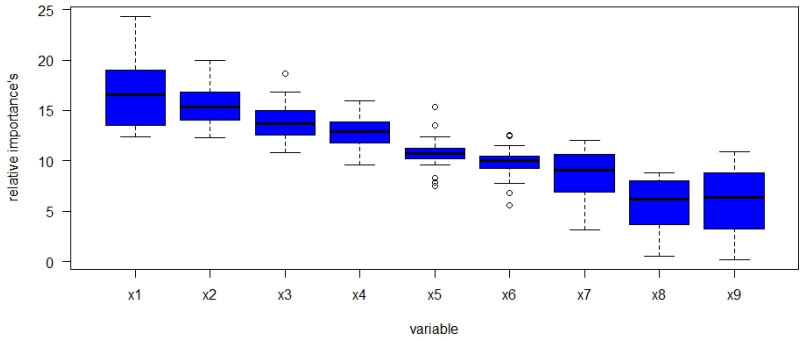
In Figure 9.a is the reference relative importance when there is no correlation and mislabeling probability. From x5 to x6 there is sudden decrease in the relative importance. Therefore, when the sample size is small the contribution methods do not work properly.

In Figure 9.b the correlation of 0.3 affected in different way each one of the variables. The variability of the relative importance of some variables increased. As in the previous figure there is a sudden decrease in the relative importance from x5 to x6. It is seems that correlation made some trends more visible like the previously stated.

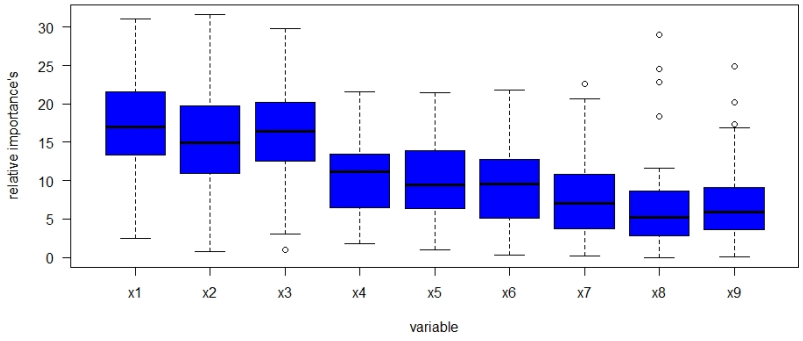
In Figure 9.c the mislabeling probability of 0.3 affected each predictor variable in a different way as correlation. But in this case the variability increased more than in the other two cases. The relative importance of some variables increases and in others decreased.



(a) Relative importance



(b) Relative importance when correlation is 0.3



(c) Relative importance when the mislabeling probability is 0.3

Figure 10: CW method with sample size 1680

In Figure 10.a the reference relative importance is given by the CW on 1680 observations with no correlation and no mislabeling probability. The variability of all the variables is similar. There is more clearer ladder than when the sample size was 85. Therefore, as the sample size increases the relative importance is better calculated.

In Figure 10.b the correlation of 0.3 increased the variability of some variable contributions while others where decreased. The ladder was still visible. Therefore, the rank of the variables was unaffected.

In Figure 10.c the mislabeling probability of 0.3 increased the variability in all the variables. It is harder to observed the ladder than in the other two previous figures. It seems that mislabelling affects more the methods than the correlation. Small sample size is affected by both factors while large sample size is more influenced by mislabelling.

	Sample size 85				Sample size 510				Sample size 1680			
	CW		Pad		CW		Pad		CW		Pad	
V	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra
x1	19.29	1	19.15	1	20.84	1	17.47	1	18.83	1	19.26	1
x2	16.52	2	16.32	2	18.97	2	17.42	2	16.91	2	17.07	2
x3	15.68	3	15.30	3	14.03	3	15.11	4	15.67	3	15.50	3
x4	13.43	4	12.69	4	13.97	4	15.30	3	12.81	4	12.34	4
x5	12.10	5	11.72	5	10.49	5	11.17	5	10.77	5	11.12	5
x6	6.92	6	6.98	6	8.34	6	8.41	6	8.63	6	9.16	6
x7	6.64	7	6.77	7	5.82	7	7.11	7	7.24	7	7.15	7
x8	6.13	8	6.48	8	4.89	8	5.10	8	5.55	8	5.14	8
x9	3.30	9	4.58	9	2.65	9	2.91	9	3.60	9	3.27	9

Table 4: Derive importance with no correlation and no mislabeling

As the sample size increases the variable rank almost remain the same. The only method affected was Pad with 510 observations. The variable contribution changes depending on the method and the number of observations. To understand the following table the next notation is introduced No Correlation and no probability of Mislabeling (NCM), Correlation (C) and probability of Mislabeling (M). Therefore, C0.1 and M0.1 mean Correlation of 0.1 and probability of Mislabeling

of 0.1.

	85		510		1680	
	CW	Pad	CW	Pad	CW	Pad
NCM	0	0	0	0.22	0	0
C0.1	0.88	0.88	0.22	0.22	0	0
C0.3	1.33	1.55	0	0.22	0.22	0.22
C0.5	0.88	0.88	0.44	0.66	0	0
M0.1	1.33	0.88	0.66	0.22	0	0
M0.2	1.33	1.11	0.66	1	0	0.66
M0.3	1.55	1.11	1.77	1.11	0.22	1.11

Table 5: Average absolute value of the difference of ranks with respect to the real rank

When the correlation was 0.1 the approaches using 85 and 510 were affected. Moreover, 85 was more affected by this factor than 510. Additionally, correlation of 0.1 and 0.3 have little impact on the rank when the sample size was 510 and 1680. Therefore, as the sample size increases this factor has less impact on the variable rank. Another finding was that when this factor was 0.3 the variable importance ranks with 1680 observations were affected. This could be due to outliers that shift the mean of the set of 50 ANNs.

Moreover, when the sample size was 85 the correlation of 0.3 affected more the techniques than 0.5. Maybe this could be due to a shift in the mean caused by outliers. Therefore, when dealing with a small sample size the variable importance methods produce unexpected results.

When the correlation in the data is controlled, both methods are affected in a similar way. In the opposite case, when the relationship in the data has a wide range of values both approaches are influenced in different ways. Therefore, it is hard to understand how these techniques are affected when the correlation is uncontrolled or bigger than 0.5. The correlation in the real data set is uncontrolled. This could be the reason why both methods were influenced in different ways.

Additionally, similar results to Paliwal and Kumar (2011) were obtained. They found that in sample size medium and large the correlation the variable rank was unaffected. In this work, it was found that the medium size data set of 510 ob-

servations was affected when the correlation was 0.5. They used three different ANNs. All the architectures had four input and one output neuron. The hidden processing units took values of one, three and six. They used the number of predictor variables to define what is a small, medium and big sample size. In that case, there were nine input neurons and probably, for this reason the correlation of 0.5 had an effect on the sample size of 510. In the present work, an additional procedure to the previous research was included that is the mislabeling probability.

Another factor that can influence the variable contribution and rank is the mislabeling probability. When this factor was 0.1 it affected more small sample size information than data with medium and big sizes. Subsequently, when this probability increased to 0.2 all the ranks with the exception of the CW approach with a sample size of 1680 were modified. And when the mislabeling probability was 0.3 all the methods in the sample sizes were influenced. Therefore, as this probability increases the methods are more influenced.

Moreover, each rank was affected in a different way by this mislabeling error. In general, this factor influenced more CW than Pad. A possible reason is that the Pad and CW methods work in different ways. Another reason it could be that as the sample size increases the final models are different.

When the number of observations was 1680 and the mislabeling probability was 0.3 the Pad technique was more affected than CW. A possible reason for this could be the bias are wrong and this causes the approach to fail. Another reason is that Oña and Garrido (2014) found that Pad method has higher variability than CW. Further research most be done to determine the causes of the previous result.

The importance in medical sets of this factor is that sometimes a doctor can misclassify a patient and this lead to false negatives or positives. Therefore, determining how the variable importance methods are affected is useful.

To solve the problems of correlation and mislabeling increasing the sample size can be done. It seems that the first feature in the data affects less than mislabeling. When the correlation was 0.5 and the sample size 1680 the rank of the variables almost did not change. But in the case of the probability of mislabeling of 0.3 most variables were in the wrong rank. The variability caused by mislabel-

ing is bigger than in the case of correlation. This indicates that as misclassification increases the variability in the contribution and rank of the variables increment. Therefore, the mislabeling probability has a bigger impact than correlation.

Mislabelling affected more the methods probably because the output vector has mistakes and it was employed to calculate the error of the model. Put differently, the optimization process was being performed in a wrong direction. As this previous metric decreased the weights were updated in a wrong way. Another reason is the values of these parameters were used for both techniques. As the sample size increased the techniques were able to decrease the variability of the variable contribution and rank.

In this work, it was discovered that adjusting the size of the simulated data have little effect on the variable contribution and almost no effect on the rank. This means that if all the data employed have the same distribution you could require only 85 observations and perform the variable importance analysis on the ANN model. The fact that the sample size is 10 or 30 times more extensive is not going to affect the results. Another important finding in Olden *et al.* (2004) is that the performance of all methods was found to slightly increase with larger sample sizes. The value of 0.22 in the Pad method with 85 observations can be due to outliers that shift the mean of the average contribution.

The results demonstrated that the probability distribution of each variable relative importance is different. In each method, the same predictor variable has a different distribution. Some reasons are the instability, mislabeling, correlation and small sample size. The distribution was more affected by mislabeling than by the other two factors. Correlation has a bigger impact than the sample size. Changing the number of observations has almost no impact on the distributions. And probably this impact can be caused by instability.

Several authors have analyzed the problem of variable importance, but they failed to achieve a consensus. An expert is required to determine the variable importance (Oña and Garrido 2014). The two approaches above have high variability not only due to the method applied but also because each time the ANN architecture is initialized with different random weights. The results of this dissertation are similar to other research papers. The difference between the present

dissertation and all the other literature reviewed is the comparison among sample size, correlation and the mislabeling probability using two techniques.

4.2.2 Comparison of performance among functions

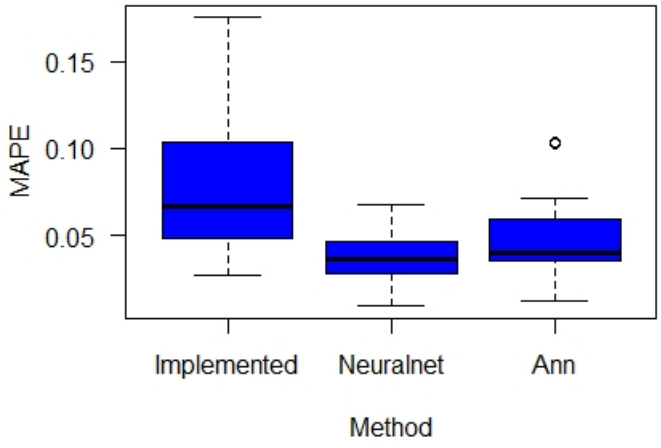
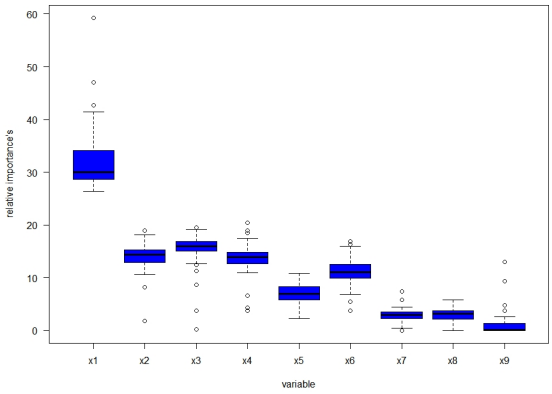


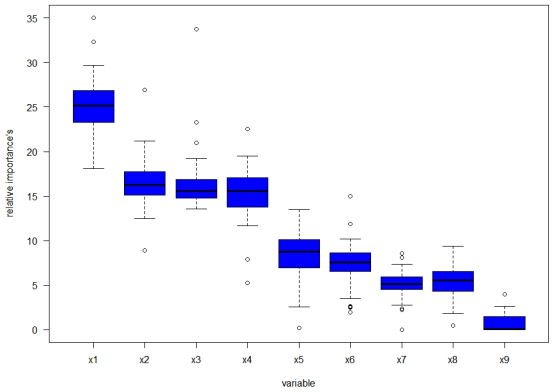
Figure 11: Performance of different functions used to implement neural networks

The function that has the best performance is Neuralnet and the worst is Implemented. The first function has more variability than the others. Ann has more variability than Neuralnet. The function Neuralnet has a higher accuracy than Ann and the Implemented algorithm. The R code taken from Tychobra (2018) produces neural networks with higher variability than the built-in R functions.

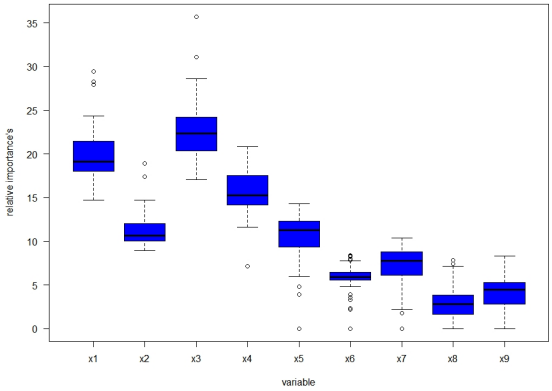
4.2.3 Implemented method to determine variable importance



(a) Relative importance when the sample size is 85



(b) Relative importance when the sample size is 510



(c) Relative importance when the sample size is 1680

Figure 12: Improve stepwise a method changing the sample size with no correlation and no mislabeling probability

The previous method is affected by the sample size. In each plot the variables relative importance changes. There are outliers in the three plots.

	85		510		1680	
	Av	Ra	Av	Ra	Av	Ra
x1	32.09	1	25.09	1	19.88	2
x2	13.92	3	16.41	2	11.42	4
x3	15.35	2	16.30	3	22.83	1
x4	13.59	4	15.30	4	15.63	3
x5	6.81	6	8.38	5	10.53	5
x6	11.21	5	7.24	6	5.74	7
x7	2.92	8	5.10	8	7.02	6
x8	2.96	7	5.32	7	2.87	9
x9	1.13	9	0.85	9	4.07	8

Table 6: Derived importance using improved stepwise a with no correlation and no mislabelling

As the sample size increases the rank and contribution of the variable changes. The least affected rank was when the sample size is 510. The approach of stepwise improved a was unsuccessful in ranking the variables. As the sample size increases this rank changes. The results were not good which is why I did not try the method in the more complicated cases with mislabelling and correlation.

5 Conclusions

The correlation and probability distribution of the data set was found. The real data set had a high variety of correlations among the variables, between Insulin and HOMA is high and between Age and BMI is low. Additionally, each of the variables have a different distribution that makes ANNs useful.

Changing the number of neurons and hidden layers and observing the behaviour of MAPE the ideal architecture was found. After implementing three different types of MLP with one, two and three hidden layers. It was determined that the best MLP consisted of a single hidden layer with five units. This metric has a lower value when using the training set than when employing the test set and this is due to overfitting. This problem should be solved in order for the model to be employed for prediction. More data should be used to avoid overfitting in the ANNs. Additionally, from the three types of ANN implementations the one with the highest accuracy is Neuralnet.

These two methods have similar contribution and the same ranking in some cases for simulated data. In the case of real data, these two techniques produced a different order of importance with the exception of the first two variables. Both approaches were affected by instability. Also these techniques were influenced by outliers. A possible way to deal with instability is removing outliers. This will diminish the variability in the two methods. The Pad method had more variability than CW. Moreover, both approaches are good determining the variable importance. Both techniques find the that Glucose and Resistin were the two most important variables.

Three types of collections of data sets were simulated with controlled sample size, correlation and mislabelling probability. From this data it was found that the variable contribution and rank was almost unaffected by the sample size. The other two factors influenced more the variable importance approaches. The correlation impacts both methods in a similar way while the mislabeling probability affects both approaches in different ways. The mislabelling influenced the most the two techniques.

These two factors influenced less the techniques as the sample size increases. The approaches using large sample sizes were influenced by misclassification but

are almost unaffected by correlation. Additionally, the instability and outliers also affected the methods using simulated data.

The rank was correctly estimated by these procedures. The methods applied to small sample size (85 data points) were more influenced by correlation and mislabelling. As the sample size increased to 1680 observations the techniques were less affected by the two factors. The techniques performed similar in simulated data while in real data the results were different.

This dissertation contributed to extend the knowledge gained in Paliwal and Kumar (2011) were only the correlation and sample size were changed. In some of the works where the methods of variable importance are compared like Olden *et al.* (2004), Mouton *et al.* (2010) and Oña and Garrido (2014) only ANN with one hidden layers were implemented. In this work it was found what happen if this number increases.

In future works it is suggested to employ larger real and simulated data sets. More than 50 ANN should be used. Monte Carlo simulations (10,000) used in Hattab *et al.* (2013) can be implemented to determine statistical distribution response functions from each of the predictor variables. The HIPR used in Kemp (2007) and the purposed method based on the interquartile range of the weights in Paliwal and Kumar (2011) were found to be better than the CW method. Therefore, these methods should be employed in future works. Another important factor that can be examined is how an error in an input variable affect the rank and contribution of the variable.

6 References

Shah, T. (2017). *About Train, Validation and Test Sets in Machine Learning*, available: https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7 [accessed 30 Aug 2019, 14:22].

De Oña, J. and Garrido, C. (2014). Extracting the contribution of independent variables in neural network models: a new approach to handle instability. *Neural Computing and Applications*, 25(3-4), 859–869. doi:10.1007/s00521-014-1573-5

Sharma, S. (2017) *Activation Functions in Neural Networks*, available: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6 [accessed 30 Aug 2017, 14:22]

Hattab, N., Hambli R., Motelica-Heino, M. and Mench, M. (2013). Neural network and Monte Carlo simulation approach to investigate variability of copper concentration in phytoremediated contaminated soils. *Journal of Environmental Management*, Elsevier, 129, pp.134-142.10.1016/j.jenvman.2013.07.003. insu-00852439

Li, L., Lu, Z., and Zhou, C. (2011). Importance analysis for models with correlated input variables by the state dependent parameters method. *Computers and Mathematics with Applications*, 62(12), 4547–4556. doi:10.1016/j.camwa.2011.10.034

Denœux, T. (2017). Computational statistics, Lecture 1: Optimizing smooth univariate functions

Masoud, N. (2014) .Predicting Direction of Stock Prices Index Movement Using Artificial Neural Networks: The Case of Libyan Financial Market. *British Journal of Economics, Management and Trade.* 4(4):597-619.

Rolnick, D., Veit, A., Belongie, S. and Shavit, N. (2017). Deep Learning is Robust to Massive Label Noise.

Shahabi, H., Khezri, S., Ahmad, B. B., and Zabihi, H.. (2012). Application of Artificial Neural Network in Prediction of Municipal Solid Waste Generation

(Case Study: Saqqez City in Kurdistan Province). World Applied Sciences Journal. 20. 336-343. 10.5829/idosi.wasj.2012.20.02.3769.

Rodriguez, J.M.O.,Mendez, C.G., Blanco, M.R.M., Tapia, S.C., Lucio, M.M., Martinez, R.J., Sanchez, L.O.S., Fierro, M.L. Martinez, Veloz, I.G., Galvan, J.C.M. and Garcia, J.A.B., (2017). Breast Cancer Detection by Means of Artificial Neural Networks. *IntechOpens*, available: 10.5772/intechopen.71256.

Traoré, B.B., Kamsu-Foguem, B. and Tangara, F. (2018). Deep convolution neural network for image recognition. *Ecological Informatics*, Elsevier, 48, pp.257-268.10.1016/j.ecoinf.2018.10.002. hal-02053205

Kim, S. and Kim, H. (2016). A new metric of absolute percentage error for intermittent demand forecasts. *International Journal of Forecasting*, 32(3), 669–679. doi:10.1016/j.ijforecast.2015.12.003

Wikipedia (2019) *Artificial neuron*, available:
$https://en.wikipedia.org/wiki/Artificial_neuron$ [accessed 30 Aug 2017, 14:22].

Patrini, G., Rozza, A., Menon, A. K., Nock, R., and Qu, L. (2017). Making Deep Neural Networks Robust to Label Noise: A Loss Correction Approach. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*
doi:10.1109/cvpr.2017.240

Sebastiani, F., Cortesi, L., Sant, M., Lucarini, V., Cirilli, C., De Matteis, E. and Federico, M. (2016). Increased Incidence of Breast Cancer in Postmenopausal Women with High Body Mass Index at the Modena Screening Program. *Journal of Breast Cancer*, 19(3), 283. doi:10.4048/jbc.2016.19.3.283

Lek, S., Beland, A., Dimopoulos, I., Lauga, J., Moreau, J. (1995). Improved estimation, using neural networks, of the food consumption of fish populations. *Mar Freshw Res* 46(8):1229–1236

Lek, S., Delacoste, M., Baran, P., Dimopoulos, I., Lauga, J. and Aulagnier, S. (1996). Application of neural networks to modeling nonlinear relationships in ecology. *Ecol Model* 90:39–52

Lek, S., Beland, A., Baran, P., Dimopoulos, I. and Delacoste, M. (1996). Role of some environmental variables in trout abundance models using neural networks. *Aquat Living Resour* 9:23–29

Dimopoulos, Y., Bourret, P. and Lek, S. (1995). Use of some sensitivity criteria for choosing networks with good generalization ability. *Neural Process Lett* 2:1–4

Kemp, S.L., Zaradic, P. and Hansen, F. (2007). An approach for determining relative input parameter importance and significance in artificial neural networks. *Ecol Model* 204:326–334

Shah, R., Rosso, K., and Nathanson, S.D. (2014). Pathogenesis, prevention, diagnosis and treatment of breast cancer. *World Journal of Clinical Oncology.* 5(3): 283–298. available doi: 10.5306/wjco.v5.i3.283

Bruning P.F.,Bonfrer J.M.G., Noord, P.A.H, Hart A.A.M, Bakker M.J. and Nooije, W.J. (1992). Insulin resistance and breast-cancer risk. *International Journal of Cancer*, 52(4):511-6, available doi: 10.1002/ijc.2910520402

Cannings, I.T., Fan, Y. and Samworth, J.R. (2019). Classification with imperfect training labels.

Luque, R., López, S.L.M., Villa, O. A. M. Luque, Isabel L. Santos-Romero, Ana Yubero-Serrano, Elena Cara-García, María Álvarez-Benito, Marina López-Miranda, José Gahete, Manuel and Castaño, Justo. (2015). Breast cancer is associated to impaired glucose/insulin homeostasis in premenopausal obese/overweight patients. *Oncotarget.* 8. doi: 10.18632/oncotarget.20399.

Drakos, D. (2018). *How to select the Right Evaluation Metric for Machine Learning Models: Part 1 Regression Metrics.* available: https://towardsdatascience.com/how-to-select-the-right-evaluation-metric-for-machine-learning-models-part-1-regrression-metrics-3606e25beae0

Artac, M. and Altundag, K. 2012. Leptin and breast cancer: an overview. *Medical Oncology.* 29:1510–1514, avaliable doi: 10.1007/s12032-011-0056-0

Assiri, A.M.A., Kamel, H.F.M. and Hassanien, M.F.R.. (2015). Resistin, Visfatin, Adiponectin, and Leptin: Risk of Breast Cancer in Pre- and Postmenopausal Saudi Females and Their Possible Diagnostic and Predictive Implications as Novel Biomarkers.*Hindawi Publishing Corporation.* available: http://dx.doi.org/10.1155/2015/253519.

Dutta P. Sarkissyan, M.· Paico, K., Wu, Y. and Vadgam J.V. (2018). MCP-1 is overexpressed in triple-negative breast cancers and drives cancer invasiveness and metastasis. *Breast Cancer Research and Treatment*, available: https://doi.org/10.1007/s10549-018-4760-8

Crisóstomo, J., Matafome, P., Santos-Silva, D., Gomes, A. L., Gomes, M., Patrício, M., Seiça, R. (2016). Hyperresistinemia and metabolic dysregulation: a risky crosstalk in obese breast cancer. *Endocrine*, 53(2), 433–442. doi:10.1007/s12020-016-0893-x

Ragab, D. A., Sharkas, M., Marshall, S., and Ren, J. (2019). Breast cancer detection using deep convolutional neural networks and support vector machines. *PeerJ*, 7, e6201. doi:10.7717/peerj.6201

American Cancer Society.2017. Breast Cancer Facts and Figures 2017-2018. *Atlanta: American Cancer Society*, Inc.

Hastie, T., Tibshirani, R. and Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition. *Springer Series in Statistics.*

WallStreetMojo.(n.d.). Correlation Coefficient Formula. available: https://www.wallstreetmojo.co coefficient-formula/ [accessed 30 Aug 2019, 14:22]

Patrício, M., Pereira, J., Crisóstomo, J., Matafome, P., Gomes, M., Seiça, R. and Caramelo, F. (2018). Using Resistin, glucose, age and BMI to predict the presence of breast cancer. *BMC Cancer*, 18(1). doi:10.1186/s12885-017-3877-1

Lo, J. Y., Baker, J. A., Kornguth, P. J., Iglehart, J. D., and Floyd, C. E. (1997). Predicting breast cancer invasion with artificial neural networks on the

basis of mammographic features. *Radiology*, 203(1), 159–163. doi:10.1148/radiology.203.1.9122385

Patel, P., Nandu, M. and Raut, P. (2019). Initialization of Weights in Neural Networks. Volume 4 | Issue 2. 73 - 79.

Tariq, N. (2018) Breast Cancer Detection using Artificial Neural Networks. *J Mol Biomark Diagn* 9: 371. doi: 10.4172/2155-9929.1000371

Li. H, Giger, M.L., Huynh, B.Q. and Antropova, N.O. (2017). Deep learning in breast cancer risk assessment: evaluation of convolutional neural networks on a clinical dataset of full-field digital mammograms,*J. Med. Imag.*4(4), 041304,doi: 10.1117/1.JMI.4.4.041304.

Nastac, P. Jalava, M. Collan, Y. Collan, T. Kuopio and B. Back.(2004), "Breast cancer prediction using a neural network model," *Proceedings World Automation Congress*, pp. 423-428.

Capasso, I., Esposito, E., Pentimalli, F., Montella, M., Crispo, A., Maurea, N. and Giordano, A. (2013). Homeostasis model assessment to detect insulin resistance and identify patients at high risk of breast cancer development: National Cancer Institute of Naples experience. *Journal of Experimental and Clinical Cancer Research*, 32(1), 14. doi:10.1186/1756-9966-32-14

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61, 85–117. doi:10.1016/j.neunet.2014.09.003

Gevrey M, Dimopoulos, I. and Lek S. (2003). Review and comparison of methods to study the contribution of variables in artificial neural network models. *Ecol Model*. 160:249–264.

Scardi, M. and Harding, L.W. (1999) Developing an empirical model of phytoplankton primary production: a neural networks case study. *Ecol Model* 120(2–3): 213–223.

Gevrey, M., Dimopoulos, I. and Lek, S. (2003). Review and comparison of

methods to study the contribution of variables in artificial neural network models. *Ecol Model* 160:249–264

Sung, A.H. 1998. Ranking importance of input parameters of neural networks. *Expert Systems with Applications* 15,405.411.

Wittek, P. (2014). Pattern Recognition and Neural Networks. *Quantum Machine Learning*, 63–71. doi:10.1016/b978-0-12-800953-6.00006-2

Hua, Y., Zhao, Z., Li, R., Chen, X., Liu, Z., and Zhang, H. (2019). Deep Learning with Long Short-Term Memory for Time Series Prediction. *IEEE Communications Magazine*, 1–6. doi:10.1109/mcom.2019.1800155

Maier, H.R., Dandy, G.C. and Burch, M.D. (1998). Use of artificial neural networks for modelling Cyanobacteria anabaena spp. in the river Murray, South Australia. *Ecological Modelling.* 105, 257/272

Castillo, E.F., Guijarro-Berdiñas, B., Fontenla-Romero, O., and Alonso-Betanzos, A. (2006). A Very Fast Learning Method for Neural Networks Based on Sensitivity Analysis. J. Mach. Learn. Res., 7, 1159-1182.

Yeh, I.C., and Cheng, W.L. (2010). First and second order sensitivity analysis of MLP. *Neurocomputing*, 73(10-12), 2225–2233. doi:10.1016/j.neucom.2010.01.011

Kriesel, D. (2007). A Brief Introduction to Neural Networks. available: http://www.dkriesel.com

George, K., Harish, M., Rao, S., and Murali, K. (2017). Comparison of neural-network learning algorithms for time-series prediction. *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI).* doi:10.1109/icacci.2017.8125808

Zhu, X. (2007). Semi-Supervised Learning Literature Survey, Page 6. *CiteSeerX* 10.1.1.99.9681

Lu, M., AbouRizk S.M. and Hermann, U.H. (2001). Sensitivity analysis of

neural networks in spool fabrication productivity studies. *J Comput Civ Eng*; 15:299–308.

Mayo clinic, breast cancer, https://www.mayoclinic.org/diseases-conditions/breast-cancer/symptoms-causes/syc-20352470 [accessed 30 Aug 2019, 14:22]

Peprah AB, and Sarkodie YA. (2018). Accuracy of clinical diagnosis, mammography and ultrasonography in preoperative assessment of breast cancer. *Ghana Med*; 52(3): 133-139 doi: http://dx.doi.org/10.4314/gmj.v52i3.5

Boughorbel, S., Al-Ali, R., and Elkum, N. (2016). Model Comparison for Breast Cancer Prognosis Based on Clinical Data. *PLOS ONE*, 11(1), e0146413. doi:10.1371/journal.pone.0146413

Afshar, H.L, Ahmadi, M., Roudbari, M., and Sadoughi, F. (2015). Prediction of Breast Cancer Survival Through Knowledge Discovery in Databases. *Global Journal of Health Science*, 7(4). doi:10.5539/gjhs.v7n4p392

Murtagh, F. (1991). Multilayer perceptrons for classification and regression. *Neurocomputing*, 2(5-6), 183–197. doi:10.1016/0925-2312(91)90023-5.

Ganggayah M.D., Taib N.A., Har Y.C., Lio P. and Dhillon S.K.. (2019). Predicting factors for survival of breast cancer patients using machine learning techniques. *BMC Medical Informatics and Decision Making.* volume 19, Article number: 48.

Ishwaran, H., Kogalur U.B., Blackstone, E.H and Lauer, M.S. (2008). Random Survival Forests. *The Annals of Applied Statistics.* 2. doi: 10.1214/08-AOAS169.

Mazurowski, M., and Szecowka, P. (2006). Limitations of sensitivity analysis for neural networks in cases with dependent inputs. 2006 IEEE *International Conference on Computational Cybernetics.* doi:10.1109/icccyb.2006.305714.

Szecówka, P. M., Szczurek, A., and Licznerski, B. W. (2011). On reliability of neural network sensitivity analysis applied for sensor array optimization. *Sensors and Actuators B: Chemical*, 157(1), 298–303. doi:10.1016/j.snb.2011.03.066.

Bekker, A.J., and Goldberger, J. (2016). Training deep neural-networks based on unreliable labels. *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2682-2686.

Gregorutti, B., Michel, B., and Saint-Pierre, P. (2016). Correlation and variable importance in random forests.*Statistics and Computing*, 27(3), 659–678. doi:10.1007/s11222-016-9646-1.

Zeng, Xinchuan and Martinez, Tony. (2001). An algorithm for correcting mislabeled data. Intell. *Data Anal.* 5. 491-502. 10.3233/IDA-2001-5605.

Fard, F.S., Hollensen, P., Mcilory, S. and Trappenberg, T. (2017). Impact of biased mislabeling on learning with deep networks. *International Joint Conference on Neural Networks (IJCNN)*, Anchorage, AK, 2017, pp. 2652-2657. doi: 10.1109/IJCNN.2017.7966180

Pechenizkiy, M., Tsymbal, A., Puuronen, S., and Pechenizkiy, O. (2006). Class Noise and Supervised Learning in Medical Domains: The Effect of Feature Extraction. *19th IEEE Symposium on Computer-Based Medical Systems (CBMS'06)*. doi:10.1109/cbms.2006.65

Mouton, A. M., Dedecker, A. P., Lek, S., and Goethals, P. L. M. (2009). Selecting Variables for Habitat Suitability of Asellus (Crustacea, Isopoda) by Applying Input Variable Contribution Methods to Artificial Neural Network Models. *Environmental Modeling and Assessment*, 15(1), 65–79. doi:10.1007/s10666-009-9192-8

Cai, J., Zheng, P., Qaisar, M. and Luo, T. (2014). Prediction and quantifying parameter importance in simultaneous anaerobic sulfide and nitrate removal process using artificial neural network. *Environmental Science and Pollution Research*, 22(11), 8272–8279. doi:10.1007/s11356-014-3976-3

Paliwal, M., and Kumar, U. A. (2011). Assessing the contribution of variables in feed forward neural network. *Applied Soft Computing*, 11(4), 3690–3696. doi:10.1016/j.asoc.2011.01.040

Ibrahim, M.A.H., Mamat, M. and Leong, W.J. (2014). BFGS method: A new search direction. *Sains Malaysiana*. 43. 1591-1597.

Igel, C. and Hüsken, M. (2000). Improving the Rprop Learning Algorithm. *Proceedings of the Second International ICSC Symposium on Neural Computation (NC 2000)*.

Cao, M., and Qiao, P. (2008). Neural network committee-based sensitivity analysis strategy for geotechnical engineering problems. *Neural Computing and Applications*, 17(5-6), 509–519. doi:10.1007/s00521-007-0143-5.

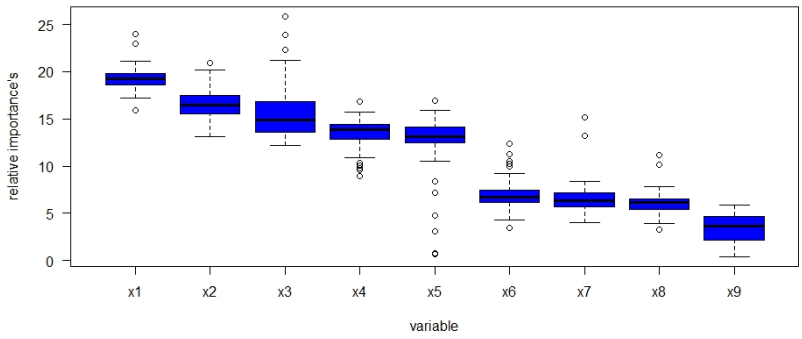
Tychobra. (2018). *How to build your own Neural Network from scratch in R*. available: https://www.r-bloggers.com/how-to-build-your-own-neural-network-from-scratch-in-r/ [accessed 30 Aug 2019, 14:22]

Rdocumentation, (n.d). *validann*. available: https://www.rdocumentation.org/packages/validann/versions/1.2.1/topics/validann [accessed 30 Aug 2019, 14:22].

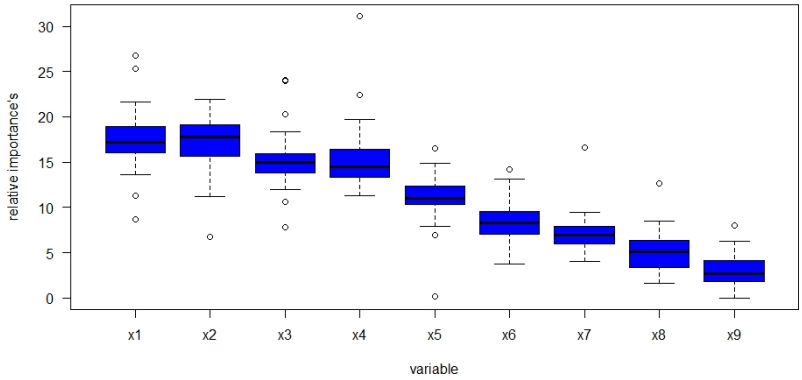
Rdocumentation, (n.d). *ann*. available: https://www.rdocumentation.org/packages/validann/versions/1.2.1/topics/ann [accessed 30 Aug 2019, 14:22].

Rdocumentation, (n.d). *neuralnet*. available: https://www.rdocumentation.org/packages/neuralnet/versions/1.44.2/topics/neuralnet [accessed 30 Aug 2019, 14:22].

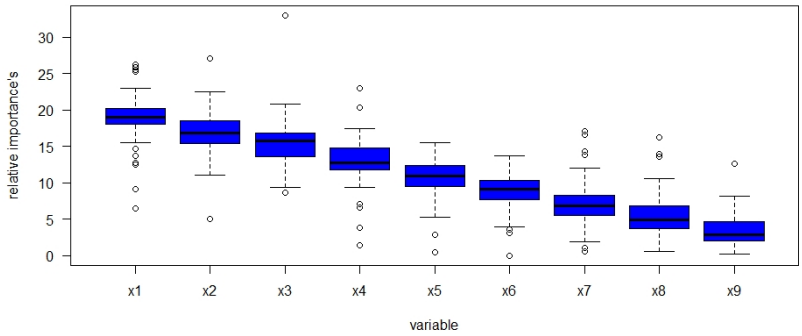
7 Annexes



(a) Relative importance when the sample size is 85

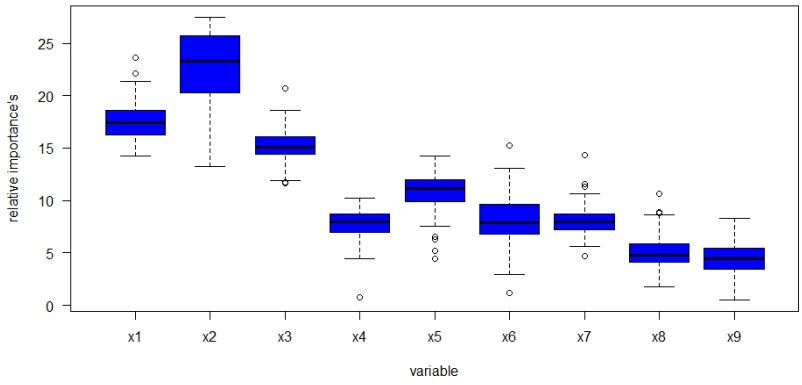


(b) Relative importance when the sample size is 510

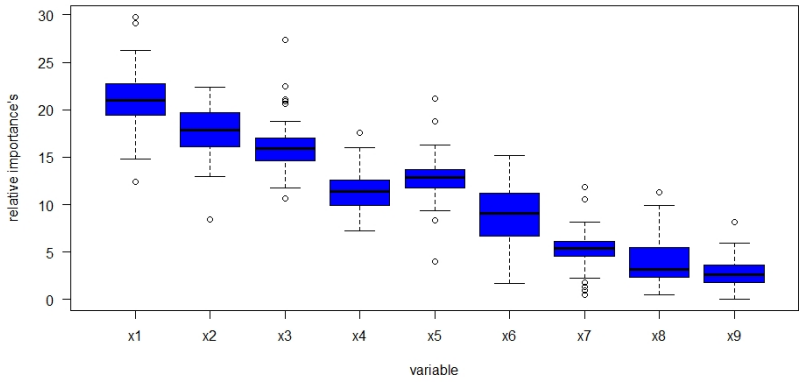


(c) Relative importance when the sample size is 1680

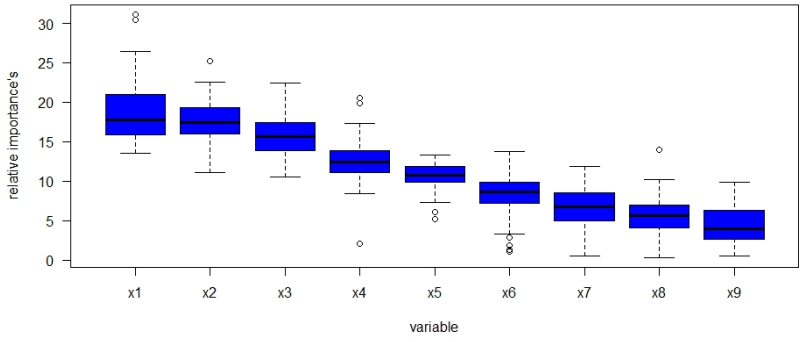
Figure 13: CW method changing the sample size



(a) Relative importance when the sample size is 85

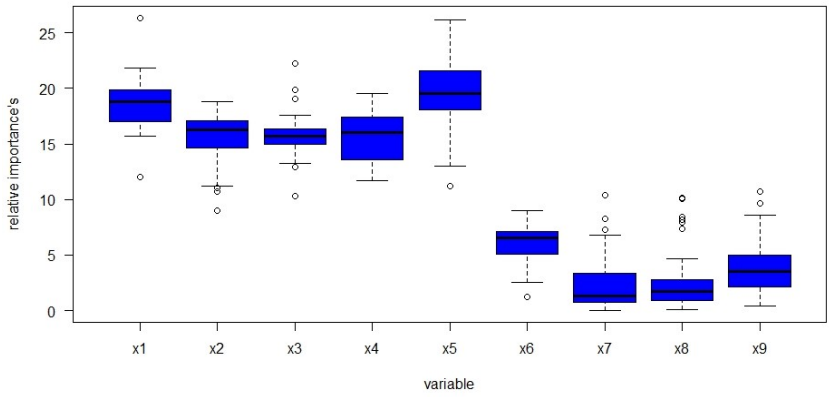


(b) Relative importance when the sample size is 510

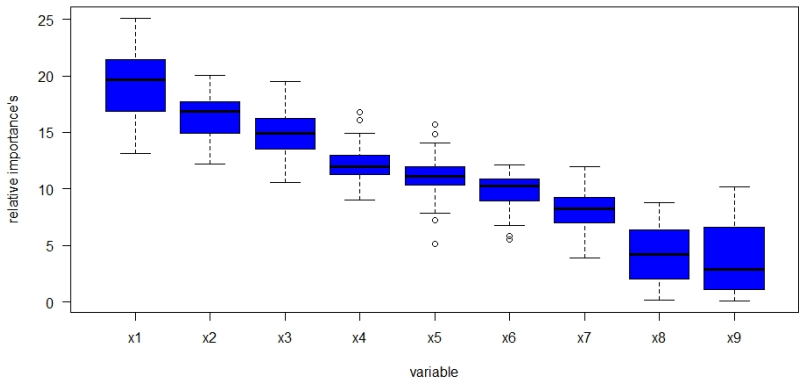


(c) Relative importance when the sample size is 1680

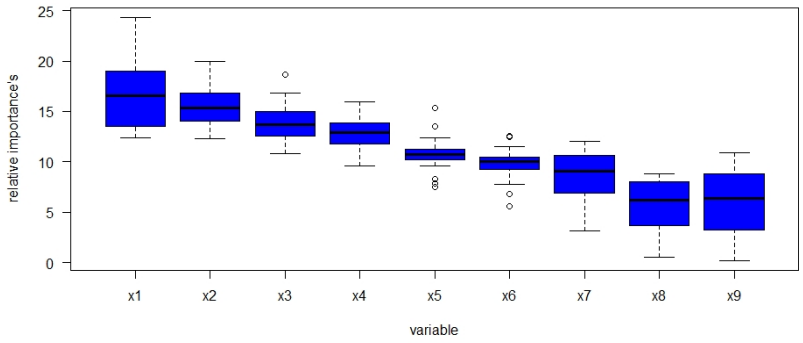
Figure 14: CW method changing the sample size with a correlation of 0.1



(a) Relative importance when the sample size is 85

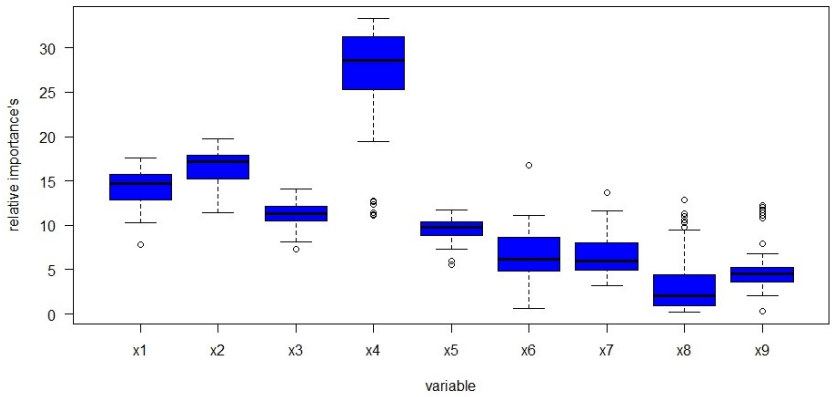


(b) Relative importance when the sample size is 510

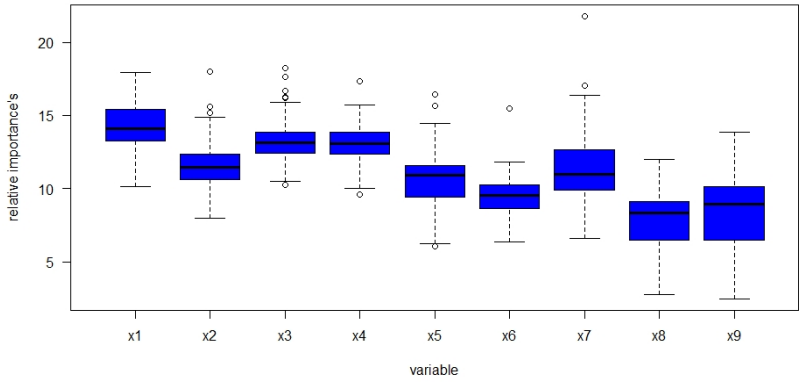


(c) Relative importance when the sample size is 1680

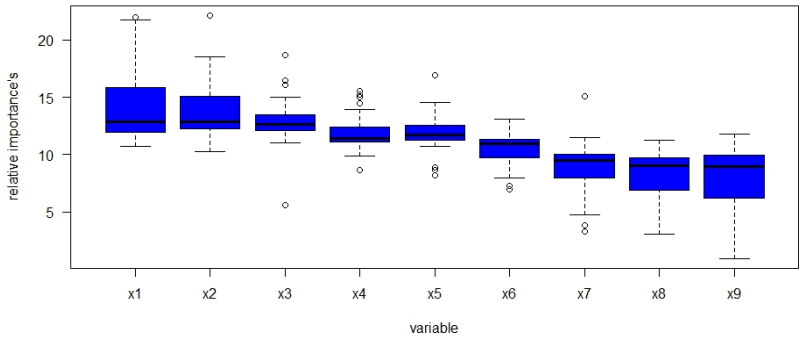
Figure 15: CW method changing the sample size with a correlation of 0.3



(a) Relative importance when the sample size is 85

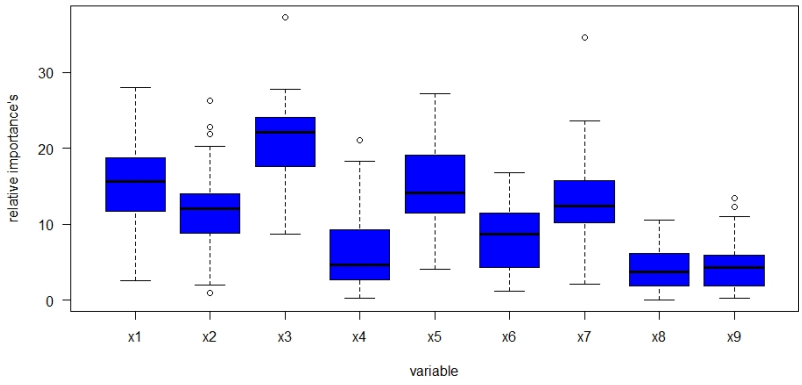


(b) Relative importance when the sample size is 510

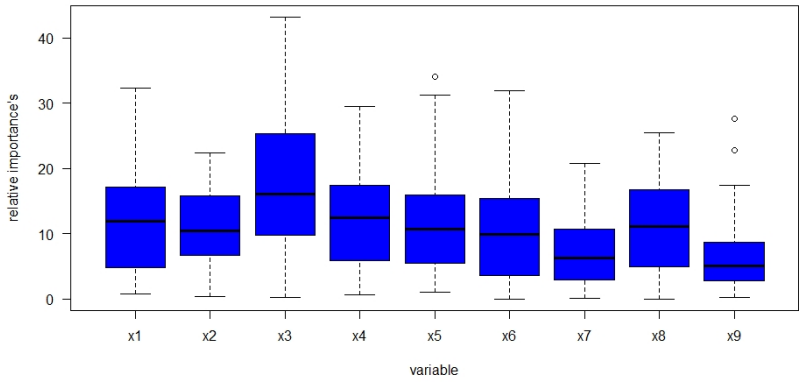


(c) Relative importance when the sample size is 1680

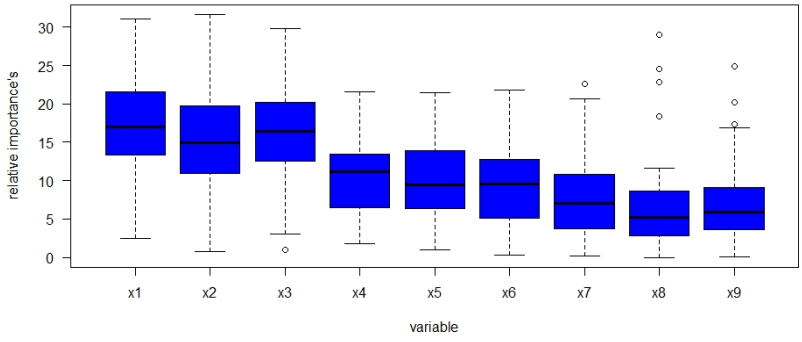
Figure 16: CW method changing the sample size with a correlation of 0.5



(a) Relative importance when the sample size is 85

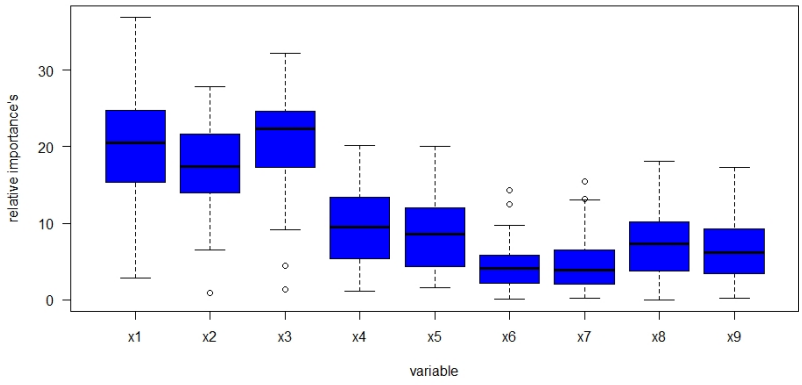


(b) Relative importance when the sample size is 510

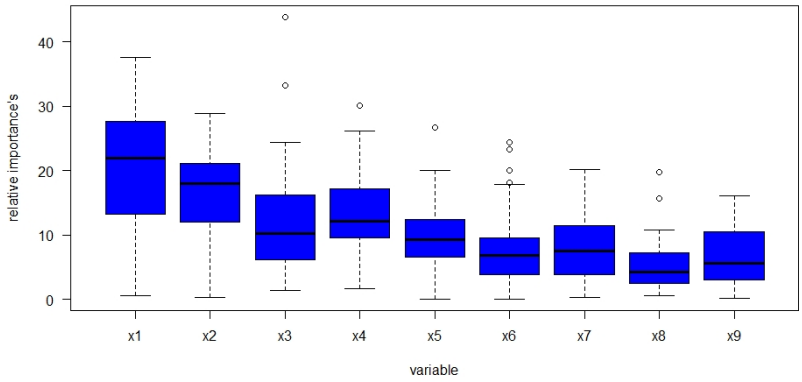


(c) Relative importance when the sample size is 1680

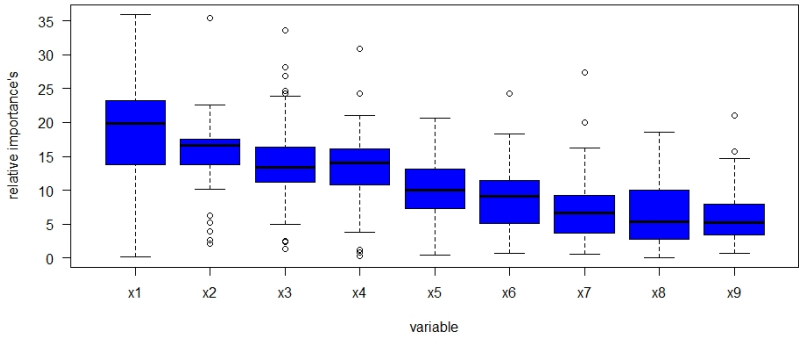
Figure 18: CW method changing the sample size with a mislabeling probability of 0.3



(a) Relative importance when the sample size is 85

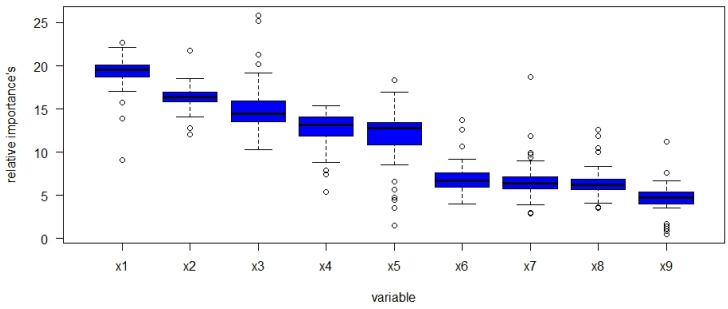


(b) Relative importance when the sample size is 510

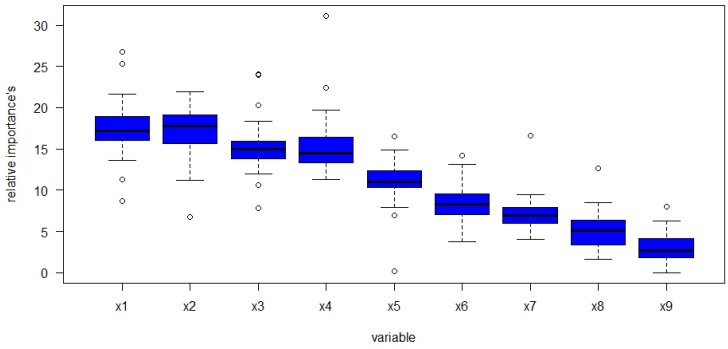


(c) Relative importance when the sample size is 1680

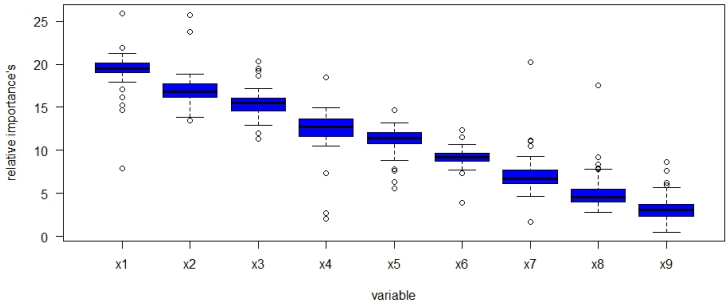
Figure 19: CW method changing the sample size with a mislabeling probability of 0.2



(a) Relative importance when the sample size is 85

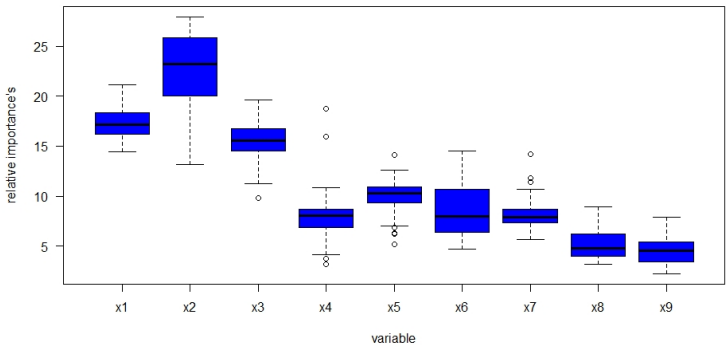


(b) Relative importance when the sample size is 510

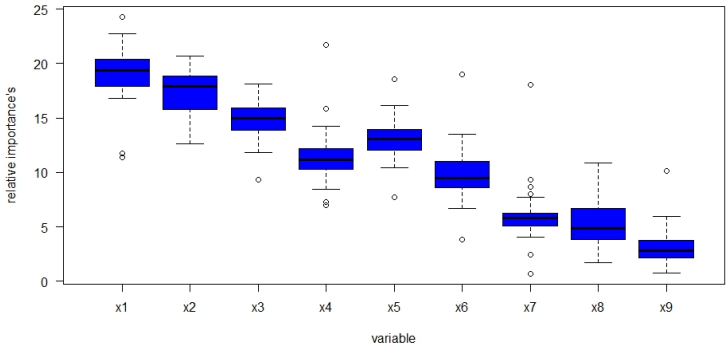


(c) Relative importance when the sample size is 1680

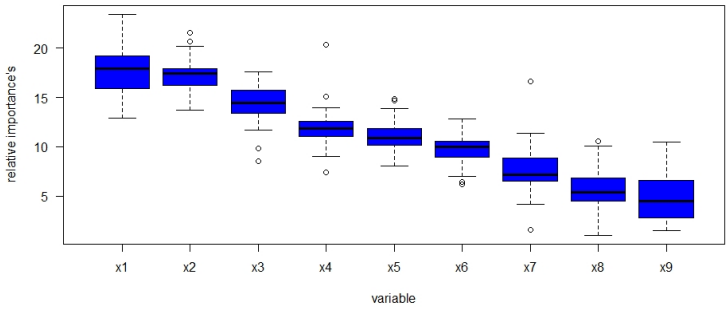
Figure 20: Pad method changing the sample size



(a) Relative importance when the sample size is 85

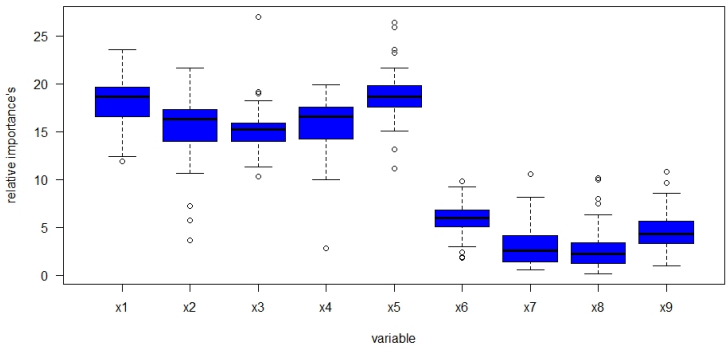


(b) Relative importance when the sample size is 510

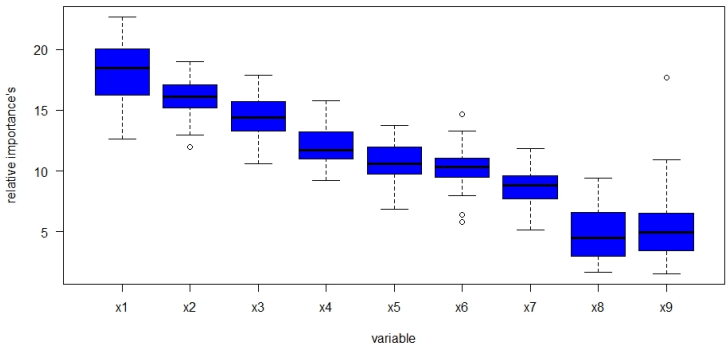


(c) Relative importance when the sample size is 1680

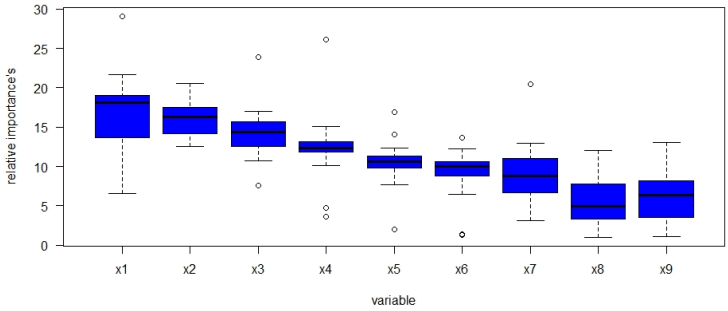
Figure 21: Pad method changing the sample size with a correlation of 0.1



(a) Relative importance when the sample size is 85

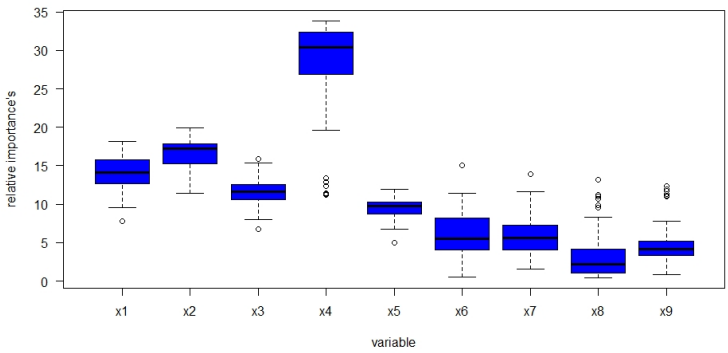


(b) Relative importance when the sample size is 510

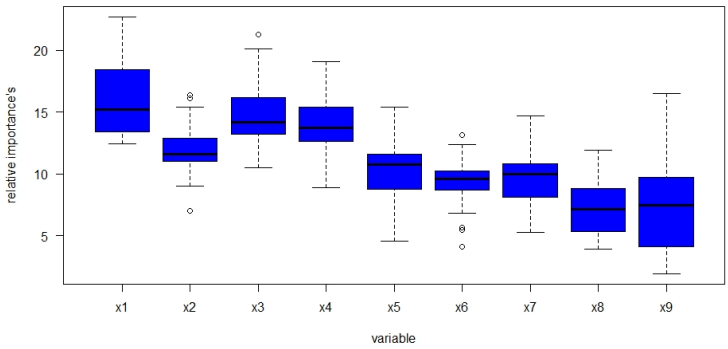


(c) Relative importance when the sample size is 1680

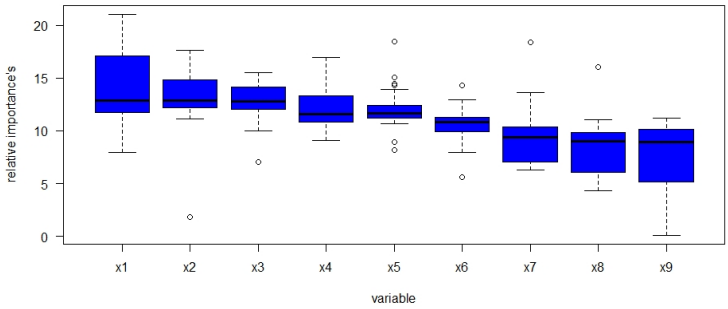
Figure 22: Pad method changing the sample size with a correlation of 0.3



(a) Relative importance when the sample size is 85

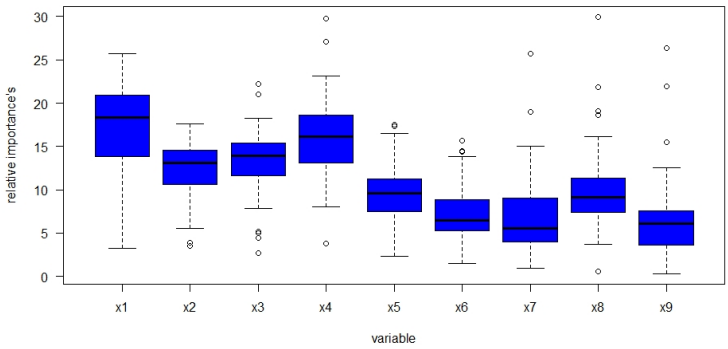


(b) Relative importance when the sample size is 510

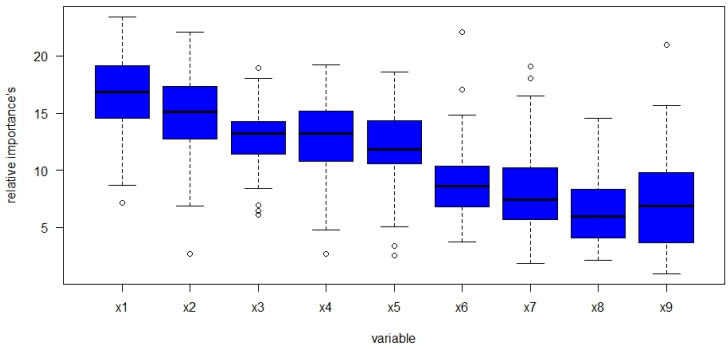


(c) Relative importance when the sample size is 1680

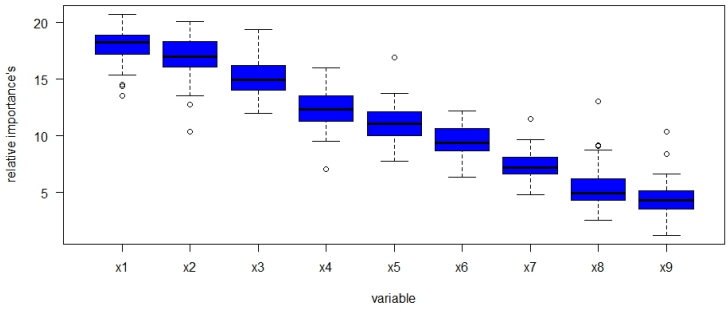
Figure 23: Pad method changing the sample size with a correlation of 0.5



(a) Relative importance when the sample size is 85

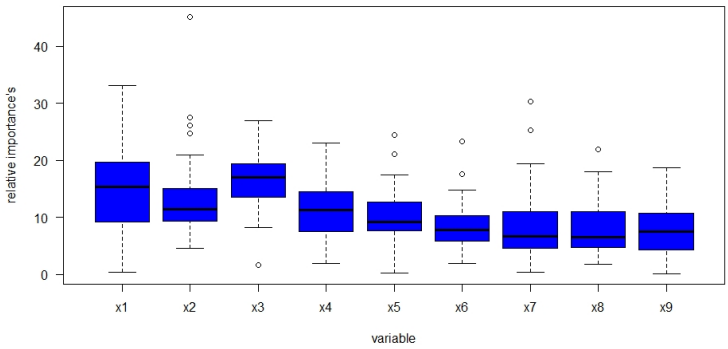


(b) Relative importance when the sample size is 510

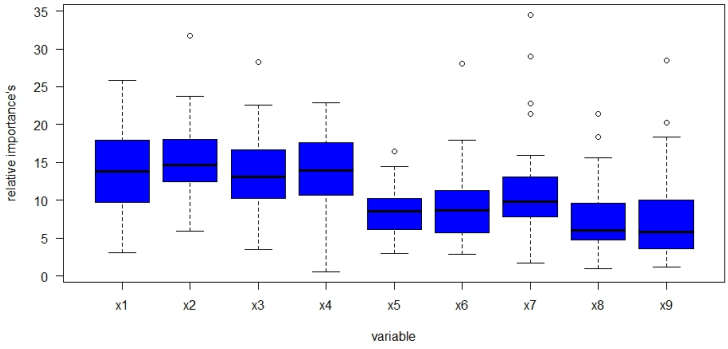


(c) Relative importance when the sample size is 1680

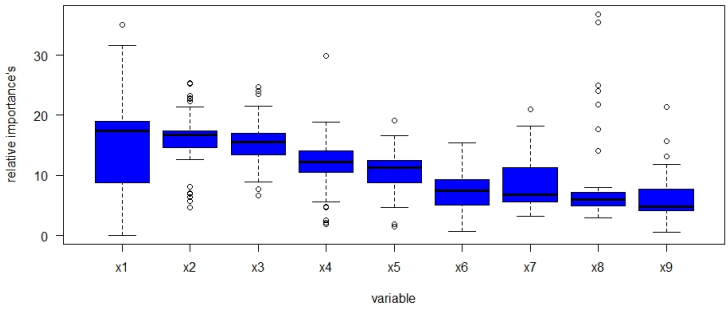
Figure 24: Pad method changing the sample size with a mislabeling probability of 0.1



(a) Relative importance when the sample size is 85

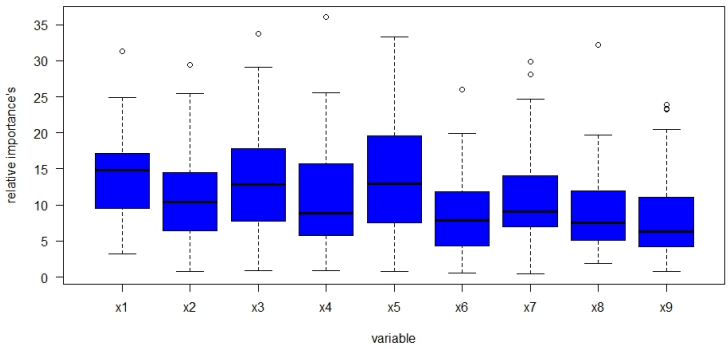


(b) Relative importance when the sample size is 510

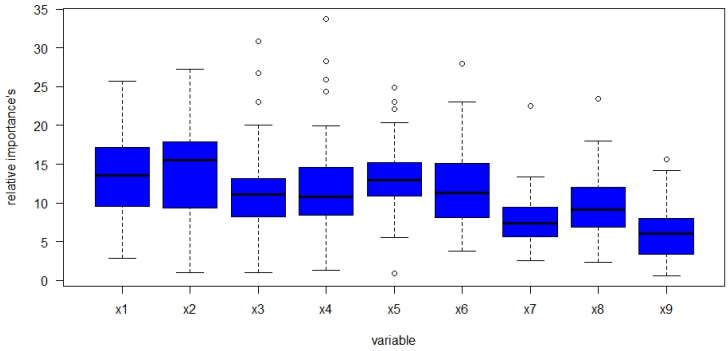


(c) Relative importance when the sample size is 1680

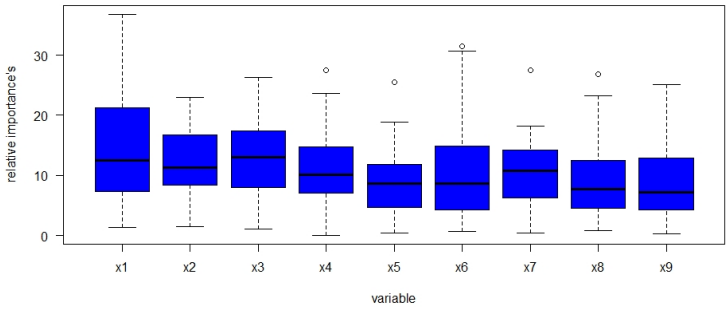
Figure 25: Pad method changing the sample size with a mislabeling probability of 0.2



(a) Relative importance when the sample size is 85



(b) Relative importance when the sample size is 510



(c) Relative importance when the sample size is 1680

Figure 26: Pad method changing the sample size with a mislabeling probability of 0.3

	Sample size 85				Sample size 510				Sample size 1680			
	CW		Pad		CW		Pad		CW		Pad	
V	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra
x1	20.26	1	20.32	1	21.16	1	19.04	1	18.95	1	17.80	1
x2	16.52	2	16.56	2	17.70	2	17.26	2	17.69	2	17.11	2
x3	12.60	3	12.52	3	16.21	3	14.83	3	15.77	3	14.33	3
x4	11.96	4	11.99	4	11.43	5	11.30	5	12.55	4	11.96	4
x5	6.92	8	6.82	8	12.75	4	13.10	4	10.66	5	10.97	5
x6	9.75	5	9.95	5	8.56	6	9.82	6	8.16	6	9.73	6
x7	7.77	7	7.72	7	5.33	7	6.00	7	6.46	7	7.61	7
x8	6.33	9	6.28	9	4.11	8	5.47	8	5.49	8	5.73	8
x9	7.89	6	7.84	6	2.75	9	3.17	9	4.27	9	4.75	9

Table 7: Derived importance with correlation of 0.1

	Sample size 85				Sample size 510				Sample size 1680			
	CW		Pad		CW		Pad		CW		Pad	
V	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra
x1	18.57	2	18.17	2	19.44	1	18.03	1	16.53	1	16.67	1
x2	15.74	4	15.51	4	16.29	2	15.94	2	15.58	2	16.04	2
x3	15.78	3	15.26	5	14.90	3	14.39	3	13.78	3	14.21	3
x4	15.62	5	15.79	3	12.20	4	11.89	4	12.90	4	12.44	4
x5	19.75	1	18.86	1	11.08	5	10.57	5	10.73	5	10.54	5
x6	6.07	6	5.83	6	9.77	6	10.28	6	9.86	6	9.64	6
x7	2.12	9	3.04	8	8.04	7	8.67	7	8.80	7	8.87	7
x8	2.51	8	2.89	9	4.29	8	4.84	9	5.82	9	5.49	9
x9	3.83	7	4.66	7	4.00	9	5.39	8	6.00	8	6.10	8

Table 8: Derived importance with correlation of 0.3

	Sample size 85				Sample size 510				Sample size 1680			
	CW		Pad		CW		Pad		CW		Pad	
V	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra
x1	14.16	3	14.01	3	16.43	1	16.14	1	14.35	1	14.39	1
x2	16.39	2	16.31	2	12.47	4	11.90	4	13.75	2	13.30	2
x3	11.33	4	11.60	4	15.49	2	14.61	2	12.83	3	12.93	3
x4	26.85	1	27.79	1	14.51	3	14.00	3	11.85	4	11.94	4
x5	9.47	5	9.41	5	10.18	5	10.35	5	11.91	5	11.98	5
x6	6.67	6	6.33	6	9.50	6	9.33	7	10.55	6	10.67	6
x7	6.65	7	6.08	7	8.79	7	9.54	6	8.97	7	9.17	7
x8	3.41	9	3.49	9	6.62	8	7.11	8	8.16	8	8.27	8
x9	5.06	8	4.99	8	6.01	9	7.01	9	7.61	9	7.35	9

Table 9: Derived importance with correlation of 0.5

	Sample size 85				Sample size 510				Sample size 1680			
	CW		Pad		CW		Pad		CW		Pad	
V	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra
x1	16.39	2	17.33	3	20.05	1	16.57	1	17.10	1	17.97	1
x2	14.71	4	12.51	2	14.64	2	14.72	2	15.67	2	17.00	2
x3	14.89	3	13.35	4	12.57	4	12.91	3	15.27	3	15.15	3
x4	17.15	1	16.26	1	12.80	3	12.73	4	11.32	4	12.22	4
x5	9.33	6	9.60	5	10.96	5	12.17	5	10.54	5	11.07	5
x6	6.55	7	7.41	6	9.31	6	9.01	6	9.54	6	9.45	6
x7	5.16	8	6.92	7	6.06	9	8.02	7	8.65	7	7.32	7
x8	10.78	5	10.04	9	6.73	8	6.64	9	6.22	8	5.41	8
x9	5.04	9	6.57	8	6.89	7	7.23	8	5.68	9	4.41	9

Table 10: Derived importance with a probability of mislabeling of 0.1

	Sample size 85				Sample size 510				Sample size 1680			
	CW		Pad		CW		Pad		CW		Pad	
V	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra
x1	19.86	2	15.69	2	20.86	1	13.93	3	18.09	1	15.57	2
x2	17.69	3	13.32	3	16.89	2	15.13	1	15.62	2	16.10	1
x3	21.12	1	16.78	1	11.98	4	13.33	4	14.31	3	15.45	3
x4	9.60	4	11.32	4	13.28	3	14.08	2	13.12	4	12.08	4
x5	8.81	5	10.11	5	9.58	5	8.41	7	9.93	5	10.60	5
x6	4.47	9	8.19	8	7.57	7	8.91	6	8.68	6	7.26	8
x7	4.95	8	8.30	7	8.07	6	11.09	5	7.33	7	8.50	6
x8	7.12	6	7.93	9	5.14	9	7.63	8	6.64	8	8.25	7
x9	6.38	7	8.36	6	6.63	8	7.49	9	6.27	9	6.21	9

Table 11: Derived importance with a probability of mislabeling of 0.2

	Sample size 85				Sample size 510				Sample size 1680			
	CW		Pad		CW		Pad		CW		Pad	
V	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra	Av	Ra
x1	15.54	2	13.93	1	12.03	3	13.53	2	17.36	1	15.28	1
x2	12.12	4	10.93	5	10.74	7	14.15	1	15.23	3	12.13	3
x3	20.98	1	13.26	3	16.90	1	11.45	6	15.94	2	12.83	2
x4	6.30	7	11.53	4	12.70	2	12.05	4	10.63	4	11.22	4
x5	15.04	3	13.39	2	11.95	4	13.38	3	10.03	5	9.02	9
x6	8.30	6	8.76	8	10.79	6	11.97	5	9.48	6	10.55	5
x7	13.10	5	10.91	6	7.10	8	7.75	8	7.91	7	10.32	6
x8	4.19	9	8.98	7	11.36	5	9.57	7	6.76	8	9.51	7
x9	4.43	8	8.31	9	6.43	9	6.14	9	6.66	9	9.14	8

Table 12: Derived importance with a probability of mislabeling of 0.3