Chapter 4: Prediction Models

4.1
Performance Metrics

The results will be compared between all data modelling techniques. For the linear regression and regression trees, the R^2 (R-square), $\overline{R^2}$ (adjusted R-square) and the Pearson´s correlation will be the measures that will indicate if the method is good or not for prediction. The $\overline{R^2}$ takes under consideration the number of input variables.

For the classification trees, there will be a 3-class evaluation ('bad', 'average' and 'good'), so the accuracy will be used. In some of the experiments, a weighted error will be used as well, computing the error 'bad'→'good' or 'good' →'bad' to be worse than the 'average'→'good' error for instance.

For the neural networks, the Pearson´s correlation will be used whenever a linear ('purelin') output neuron is used (prediction from 0 to 6) and the accuracy when three 'logsig' neurons are used as output neurons (3-class evaluation: 'bad', 'average' and 'good') as well as the weighted error. The different types of neurons are better described on chapter 4.3.2.1.1. The weighted error will be used combined with the weighted sum of probabilities from the three 'logsig' neurons, since each of them will provide as output, a number from 0 to 1, a probability that the class 'bad', 'average' or 'good' was used to evaluate the dialogue.

4.1.1
R^2 and $\overline{R^2}$

R^2, often called the coefficient of determination, is defined as the ratio of the sum of squares explained by a regression model and the total sum of squares around the mean

R^2 = 1-SSE/SST = SSR/SST, where:

$$\mathrm{SSE} = \sum_{i=1}^{n} \hat{e}_i^{\,2}$$

$$\mathrm{SST} = \sum_{i=1}^{n} \left(y_i - \bar{y}\right)^2$$

$$\mathrm{SSR} = \sum_{i=1}^{n} \left(\hat{y}_i - \bar{y}\right)^2$$

SSE is the “Sum of squares, error”, SST is the “Sum of squares, total”, SSR is the “Sum of squares, regression” and n = sample size (number of observations in the training process).

It is often referred to as the proportion of variation explained by the model, but sometimes it is called the proportion of variance explained. In sample terminology, variances are "mean squares". Thus the estimated variance of Y is ***MST = SST/(n-1)*** and the estimated residual or error variance is ***MSE = SSE/(n-p-1)*** where p is the number of predictors in the regression equation. An “average” is obtained by dividing by degrees of freedom rather than by n in order to make the sample mean squares unbiased estimates of the population variances.

Regression analysis programs also calculate an "adjusted" R^2. The best way to define this quantity is: $\overline{R^2}$ ***= 1 - MSE / MST***, since this emphasizes its natural relationship to the coefficient of determination.

While R^2 will never increase when a predictor is dropped from a regression equation (on training/in-sample data only), $\overline{R^2}$ may be larger.

Below is the traditional formula for expressing $\overline{R^2}$ in terms of the ordinary R^2. It shows explicitly the "adjustment" process, and also demonstrates that $\overline{R^2}$ is always smaller:

$$\overline{R}^2 = 1 - \frac{(n-1)}{(n-p)}\left(1 - R^2\right)$$

, where n is the sample size and p is the number of predictors.

So, the R^2 (the coefficient of determination) in theory is the percent of the Total Sum of Squares that is explained, the Regression Sum of Squares (explained deviation) divided by Total Sum of Squares (total deviation). This calculation yields a percentage. It also has a weakness. The denominator is fixed and the numerator can only increase, so therefore each additional variable used in the equation will, at least, not decrease the numerator and will probably increase the numerator at least slightly, resulting in a higher R^2 (on training/in-sample data only), even when the new variable doesn't seem to have much relation to the explained variable. The $\overline{R^2}$ value is an attempt to correct this issue, by adjusting both the numerator and the denominator according to the number of variables used.

On the linear regression, the smaller the variability of the residual values around the regression line relative to the overall variability, the better is the prediction. For example, if there is no relationship between the *X* and *Y* variables, then the ratio of the residual variability of the *Y* variable to the original variance is equal to 1.0. If *X* and *Y* are perfectly related then there is no residual variance and the ratio of variance would be 0.0. R^2 is a measure for training (in-sample) data originally, but the same formula will be used on test (out-of-sample) data for comparison purposes.

4.1.2
Pearson´s correlation

For regression models, the Pearson´s correlation coefficient is the square root of R^2, so its use is redundant. On non-linear models or using test (out-of-sample) data only, its use has a distinct meaning, so it will be used on our studies.

The Pearson's correlation coefficient, also called linear correlation coefficient is given by the formula:

$$P_{corr} = \frac{\sum_i \left(M_i - \overline{M}\right)\cdot\left(P_i - \overline{P}\right)}{\sqrt{\sum \left(M_i - \overline{M}\right)^2}\sqrt{\sum \left(P_i - \overline{P}\right)^2}}$$

, where Pi are the fitted values from the regression analysis amd Mi are the dependent variables.

Its value ranges between -1 and +1, inclusive. A value of +1, the so-called "complete positive correlation" corresponds to all the pairs (M_i,P_i) laying on a straight line with positive slope in the scatter diagram. The "complete negative correlation" corresponds to all the pairs on a straight line with negative slope, and it has a value of -1. A value of the Pearson's correlation coefficient near to zero indicates the absence of correlation between the variables.

Customarily, the degree to which two or more predictors (independent or *X* variables) are related to the dependent (*Y*) variable is expressed in the correlation coefficient *R*, which is the square root of R^2 (on training/in-sample data). In multiple regression, *R* can assume values between 0 and 1. To interpret the direction of the relationship between variables, one looks at the signs (plus or minus) of the regression or its coefficients. If the coefficient is positive, then the relationship of this variable with the dependent variable is positive ; if the coefficient is negative then the relationship is negative. Of course, if the coefficient is equal to 0 then there is no linear relationship between Y and the independent variables.

4.2
Selection of input and target variables

A correlation test between the input and target variables will be made to decide which variables are the best to be used in the experiments. The full list of correlations from both systems can be found on Chapter 7.

One of the problems with the simple correlation test to choose which variables are the most adequate to be used in prediction models, is that the independent variables may be highly correlated to each other. In theory the best predictions can be made whenever the input variables are themselves orthogonal (correlation close to zero between each other).

For the PARADISE model, an automated procedure is generally selected through a method called *forward stepwise*, that enters the predictors one by one depending on maximum reduction of the residual variance. This is again only valid for training (in-sample) data, the results on test (out-of-sample) data are not

the same and this is why the correlation test is important to select the variables that correlate the most to the target values, in order to choose correctly variables to use as input and be used as test (out-of-sample) data. This and of course the theoretical relation between input and target values will be taken under consideration.

Depending on the experiment, different variables will be used, depending on the target value that is more correlated to them and on the objective of the experiment.

Table 5 shows the main correlations from the BoRIS System:

Table 5: Highest Pearson´s correlations between input and target variables from the BoRIS System

Explanatory Variable	Question B0	Question B23	Mean Questions B
#turns	-0,341	-0,313	-0,366
CA:#IA	-0,329	-0,312	-0,354
PA:CO	-0,322	-0,282	-0,345
TS_ord	0,160	0,285	0,250
WA_iso	0,148	0,062	0,149
WPST	0,201	0,225	0,294
UCT	-0,278	-0,259	-0.303
IC	0,116	0,143	0,100
#Sys.questions	-0,315	-0,298	-0,353

These variables will be used on the experiments. See complete table 8.1 with all the correlations on Chapter 7.

The variable ‘weighted CA:IA”, created during the making of this work, has the following correlations to target values shown on table 6:

Table 6: Comparison of correlations from variables "Weighted CA:IA"/ CA:#IA and the main target values

Explanatory Variable	Question B0	Question B23	Mean Questions B
CA:#IA	-0,329	-0,312	-0,354
Weighted CA:IA	-0,223	-0,214	-0,261

This means the new variable Weighted CA:#IA has a smaller correlation than CA:#IA when compared to the main target variables. Further analysis are displayed on Chapter 6.2.

The entire list of correlations from the INSPIRE system is shown on table 8.2, Chapter 7 – Appendix. The 4 most correlated input variables are %UCT, Understood error rate, task success, WA_iso (see table 8).

Since %UCT and UER have a high correlation between each other (see table 7), only one of them will be used: %UCT.

Table 7: Correlation between %UCT and UER

		Z-Wert: underst. error rate	Z-Wert: % user correction turns
Z-Wert: underst. error rate	Pearson Correlation	1	-,745(** Sig. ,000)
Z-Wert: % user correction turns	Pearson Correlation	-,745(** Sig. ,000)	1

** Correlation is significant at the 0.01 level or less (2-tailed).

Table 8: Highest Pearson´s correlations between input and target variables from the INSPIRE System that will be used on the experiments. See complete table 8.2 with all the correlations on Chapter 7.

		%user correction turns	word accuracy	task success	overall quality
Z-Wert: % user correction turns	Pearson Correlation	1	-,386 (** Sig. ,001)	-,191 (Sig. ,119)	-,408 (** Sig. ,001)
Z-Wert: word accuracy	Pearson Correlation	-,386 (** Sig. ,001)	1	,213 (Sig. ,081)	,299 (* Sig. ,016)
Z-Wert: task success	Pearson Correlation	-,191 (Sig. ,119)	,213 (Sig. ,081)	1	,600 (** Sig. ,000)
Overall quality	Pearson Correlation	-,408 (** Sig ,001)	,299 (* Sig ,016)	,600 (**Sig ,000)	1

** Correlation is significant at the 0.01 level or less (2-tailed).

* Correlation is significant at the 0.05 level or less (2-tailed).

4.3
Modelling Techniques

4.3.1
Linear Regression

Theory

The main purpose of multiple linear regression (MLR, the term was first used by Pearson, 1908) is to learn more about the relationship between several independent or predictor variables and a dependent or target variable.

MLR attempts to model the relationship between two or more explanatory variables and a response variable by fitting a linear equation to observed data. Every value of the independent variable X is associated with a value of the dependent variable Y:

$Y = b_0 + b_1 X_1 + b_2 X_2 + \ldots + b_K X_K + u_K$, where k is the number of input variables, the b´s are the coefficients, and u_K is the error term. The higher the

coefficient is, the greater the impact of the independent variable on the dependent variable. MLR is based on least squares: the model is fit in such a way that the sum-of-squares of differences of observed and predicted values, is minimized. The performance of the method is then tested on data not used to fit the model (independent test/out-of-sample data).

The MLR model is based on several assumptions, including the following assumptions regarding the behaviour of the error term:

- $E(u_K|\underline{X})=0, \forall\ k$
-
- $E(u^2_K|\underline{X})= \sigma^2 \forall\ k$
-
- $E(u_K u_j\ |X_K\ X_j)=0 \forall\ k \neq j$

Provided the assumptions are satisfied, the regression estimators are optimal in the sense that they are unbiased, efficient and consistent. Unbiased means that the expected value of the estimator is equal to the true value of the variable. Efficient means that the estimator has a smaller variance than any other estimator within a certain class. Consistent means that the bias and variance of the estimator approach zero as the sample size approaches infinity.

One of the assumptions from Ostrom (1990, p.14) is *linearity*: the relationship between the target value and the predictors is linear. The MLR models applies to linear relationships. If relationships are nonlinear, there are two recourses: (1) transform the data to make the relationships linear, or (2) use an alternative statistical model (in this case, neural networks,
Regression and classification trees).

In summary, the purpose of multiple linear regression is to establish a quantitative relationship between a group of predictor variables and a response. This relationship is useful for understanding which predictors have the greatest effect, knowing the direction of the effect (i.e., increasing *x* increases/decreases *y*) and using the model to predict future values of the response when only the predictors are currently known.

On the PARADISE model, the regression analyses aim at predicting high-level quality aspects like 'overall user satisfaction' The target values for the models in this work were either chosen according to the classification given by the QoS taxonomy (Question B0 and Question B23) or calculated as a simple arithmetic mean over different quality aspects (Mean Questions B).

Progress so far with Linear Regression

The target variable on the PARADISE model is a "performance" measure, an estimation from user satisfaction. The model postulates that: "performance can be correlated with a meaningful external criterion such as usability, and thus that the overall goal of a spoken dialogue agent is to maximize an objective related to usability. User satisfaction ratings[…] have been frequently used in the literature as an external indicator of the usability of an agent.". This combined "performance" for a SDS is a weighted linear combination of task-based success measures and "dialogue costs", which can be of two different forms: "dialogue efficiency costs" and "dialogue quality costs". The first refers to the system's efficiency on helping the user to complete the test and the second one to other aspects of the system that may have an influence on the user's perception of the system.

An estimation of "User Satisfaction" (USw) from the PARADISE model is given by the following equation:

$$\widehat{US_w} = \alpha \cdot \mathcal{N}(\kappa) - \sum_{i=1}^{n} w_i \cdot \mathcal{N}(c_i)$$

,where κ is the task success, α is the weight of κ , *w*i is the weight for "cost" measures and *N* is the z-score normalization function which normalizes the values to a mean of zero and a standard deviation of one. "User satisfaction" should then be a sum of a "maximum task success" and "minimal dialogue costs". (For further details, see [4] and [5])

The PARADISE model uses linear regression but covers only training (in-sample) data. A stepwise multivariate linear regression procedure (Walker et al., 2000) is used to automatically select the variables to be included in the model. These results are limited in my point of view, since the selection the selection is

in-sample and it does not give rates on out-of-sample predictions (new users for instance).

Experiments will be made using the same idea from the PARADISE framework, but applying multivariate linear regression on test (out-of-sample) data, using the 'leave-one-out' method explained on Chapter 3.2, user by user.

4.3.2
Neural Networks

Theory

A neural network is an example of a nonlinear regression model. An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. ANNs learn by example. A neural network configured for a specific application, such as pattern recognition or data classification, through a learning (training) process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANNs as well.

Neural networks have the remarkable ability to map dependence structure from complicated or imprecise data. They can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyse. It can then be used to provide projections given new situations of interest.

The most common type of artificial neural network consists of three groups, or layers, of units: a layer of input units is connected to a layer of hidden units, which is connected to a layer of output units (See Figure 2). The activity of the input units represents the raw information that is fed into the network. The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units. The

behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

A feed-forward backpropagation neural network was chosen to be used for this experiment. Feed-forward Neural Networks (Figure 2) allow signals to travel one way only, from input to output. There is no feedback (loops), the output of any layer does not affect that same layer. Feed-forward Neural Networks tend to be straight forward networks that associate inputs with outputs. They are extensively used in pattern recognition. Feed-forward networks consist of Nl layers using the DOTPROD weight function, NETSUM net input function, and the specified transfer functions. The first layer has weights coming from the input. Each subsequent layer has a weight coming from the previous layer. All layers have biases. The last layer is the network output.

In practice we are often limited in the number of training patterns available, and in many applications this may indeed be a severe limitation. Note that the bias-variance problem implies that, for example, a simple linear model (single-layer network) might, in some applications involving relatively small data sets, give superior performance to a more general non-linear model (such as a multi-layer network) even though the latter contains the linear model as special case. [2]

The networks used in this work have the objective to solve classification and regression problems. On the classification problems, the goal is to obtain the probabilities of the different classes expressed as target values (or functions) from the input variables. On the regression problems, the goal is to obtain a linear value (related to the target value itself).

The performance of a network can certainly be improved by eliminating unnecessary input variables. Indeed, even input variables that carry a small amount of information should sometimes be eliminated. The selection of input variables is therefore a critical part of the neural networks design.

3-class evaluation

As seen on Chapter 3.1, for the neural networks approach, a 3-class evaluation system was created, to replace the 0 to 6 evaluation from the questionnaires used on the experiments: bad (0 to 2.4), neutral (2.5 to 3.5) and

good (3.6 to 6.0). This was made in order to use a feed-forward backpropagation neural network with an output layer with 3 ‘logsig’ neurons (figure 2) where each one of them gives a probability for the corresponding class.

‘Logsig’ neurons are neurons with a logarithmic sigmoid transfer function, ‘Purelin’ neurons are neurons with a linear transfer function and ‘Tansig’ neurons are neurons with a hyperbolic tangent sigmoid transfer function.

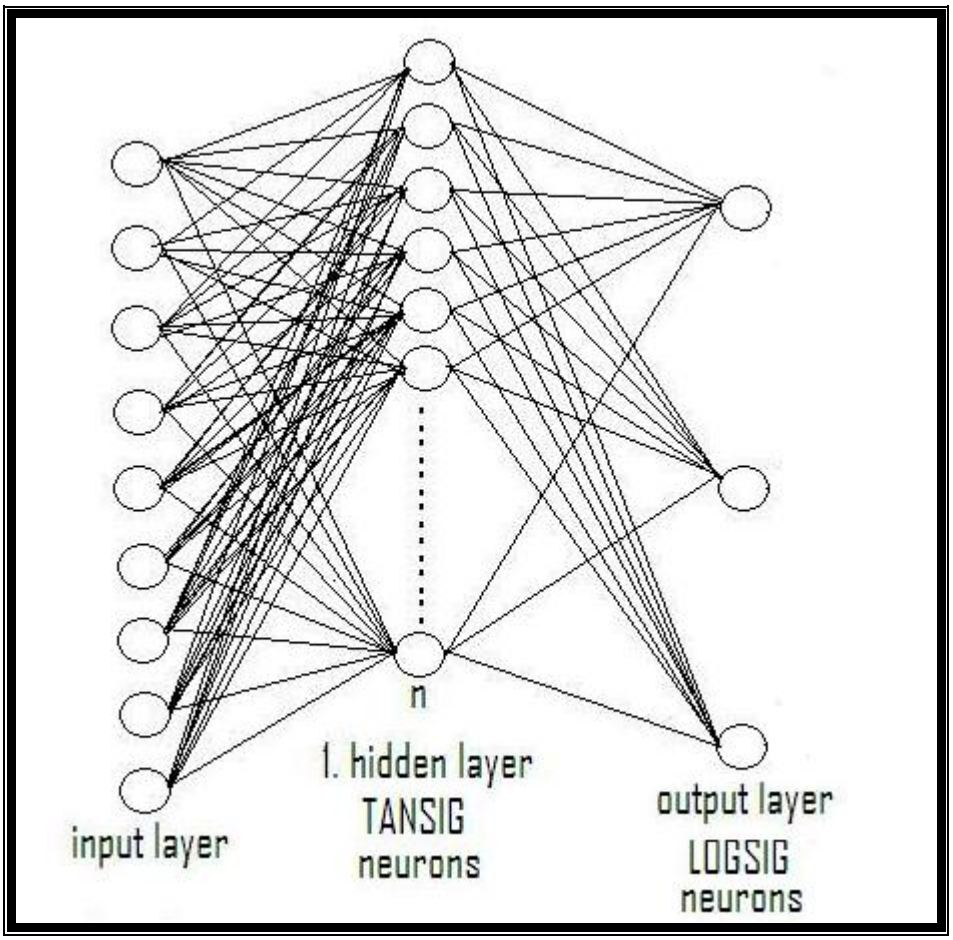

Figure 2 - Structure of a neural network with 3 ‘logsig’ neurons as output.
‘tansig’ neuron: neuron with a hyperbolic tangent sigmoid transfer function.
‘logsig’ neuron: neuron with a logarithmic sigmoid transfer function.

The other approach to be used with NNs is to use a single ‘purelin’ neuron as output, that would generate a linear value (0 to 6 on the BoRIS system case).

The Cross-validation method

When using non-linear optimization algorithms, a choice must be made regarding when to stop the training process of the NN. The possible choices are:

1-Stop after a fixed number of iterations.
2-Stop when a predetermined amount of CPU time has been used.
3-Stop when the error function falls bellow some specified value.
4-Stop when the relative change in the error function falls below some specified value.
5-Stop training when the error measured using an independent validation set starts to increase.

The choice number 5 will be used for all experiments, since it uses independent data for validation. The procedure of *cross-validation* (Stone, 1974, 1978; Wahba and Wold, 1975) will be used, since there will be maximum 200 dialogues for the BoRIS system. Here the entire data set will be divided in S distinct segments (one for each user). The training set will have S-1 segments and test the performance by evaluating the error function using the remaining segment. This process is repeated for each of the S possible choices for the segment which is omitted from the training process, and the test errors averaged over all S results. Such a procedure allows us to use a high proportion of the available data to train the network. The disadvantage of such an approach is that it requires the training process to be repeated S times which in some circumstances could lead to a requirement for large amounts of processing time.

Since the goal is to find the network having the best performance on new data, the simplest approach to the comparison of different networks is to evaluate the error function using data which is independent of that used for training. Various networks are trained by minimization of an appropriate error function defined with respect to a *training* data set. The performance of the networks is then compared by evaluating the error function using an independent *validation* set, and the network having the smallest error with respect to the validation set is selected. This approach is called the *hold out method*. Since this procedure can itself lead to some over-fitting to the validation set, the performance of the

selected network should be confirmed by measuring its performance on a third independent set of data called a *test* set. For the BoRIS System for instance, from the dialogues from 40 users, the network would be trained with the dialogues from 37 users, used the dialogues from two independent users for validation , the 'early-stop' technique, stopping the training of the network before it gets overfitted, and the dialogues from one independent user to test. This whole process would be then repeated 40 times.

Generalization

Generalization requires prior knowledge, as pointed out by Hume (1973/1978). For any practical application, it is necessary what the relevant inputs are. And it is necessary to know that the cases from the test (out-of-sample) data, the ones where the generalization will be tested, have some resemblance to the training cases. Therefore, there are three conditions that are typically necessary for good generalization:

1. The first necessary condition is that the inputs to the network contain sufficient information pertaining to the target, so that there is a mathematical function relating correct outputs to inputs with the desired degree of accuracy. It cannot be expected that a network would learn a nonexistent function. Neural networks are not a magic tool. Finding good inputs for a net and collecting enough training (in-sample) data often take far more time and effort than training the network.

2. The second necessary condition is that the function that is being searched (that relates inputs to correct outputs) could be, in some sense, smooth. In other words, a small change in the inputs should, most of the time, produce a small change in the outputs. For continuous inputs and targets, smoothness of the function implies continuity and restrictions on the first derivative over most of the input space. Some neural networks can learn discontinuities as long as the function consists of a finite number of continuous pieces. Very nonsmooth functions such as those produced by pseudo-random number generators and encryption algorithms cannot be generalized by neural networks. Often a

nonlinear transformation of the input space can increase the smoothness of the function and improve generalization. For classification, if you do not need to estimate posterior probabilities, then smoothness is not theoretically necessary. In particular, feed-forward networks with one hidden layer trained by minimizing the error rate universally consistent classifiers if the number of hidden units grows at a suitable rate relative to the number of training cases (Devroye, Györfi, and Lugosi, 1996). However, it is likely to obtain better generalization with realistic sample sizes if the classification boundaries are smoother.

Thus, for an input-output function that is smooth, if there is a test (out-of-sample) case that is close (similar) to some training (in-sample) cases, the correct output for the test case will be close to the correct outputs for those training cases. If there is an adequate sample for the training set, every case in the 'population' will be close to a sufficient number of training cases. Hence, under these conditions and with proper training, a neural network will be able to generalize reliably to the 'population'.

If there is more information about the function, like that the outputs should be linearly related to the inputs, one can take advantage of this information by placing constraints on the network or by fitting a more specific model, such as a linear model, to improve generalization. Extrapolation is much more reliable in linear models than in flexible nonlinear models, although still not nearly as safe as interpolation. Such information can also be used to choose the training cases more efficiently. For example, with a linear model, the training cases at the outer limits of the input space should be chosen instead of evenly distributing them throughout the input space.

3. In theory, a good generalization happens when the training (in-sample) cases are sufficiently large and representative of the set of all cases that are intended to be tested afterwards. This is important because there are two different types of generalization: interpolation and extrapolation. Interpolation happens when test cases are more or less surrounded by nearby training cases, everything else is extrapolation. This means the cases that are outside the range of training (in-sample) data require extrapolation. Interpolation, for the normal appliances from neural networks can often be done reliably, but extrapolation is notoriously

unreliable. So therefore, it is important to have sufficient training (in-sample) data to avoid the need for extrapolation.

Applicability and properties on quality prediction

The disadvantages of the Neural Networks technique are that the whole structure is like a black box: internal characteristics and components of the system are invisible to the evaluator. Only input output relation of the system is considered without regarding the specific mechanisms linking input to output. The advantage is that it calculates functions and algorithms that could be extremely complex by itself.

This means: neural network systems can help where one can't formulate an algorithmic solution, where one can get a big amount of examples of the behaviour required and where one need to pick out the structure from existing data.

4.3.3
Classification and Regression Trees

Theory

The classification or regression trees method is a type of regression fitting approach where one does not need to know the relationship between the target and predictors, and also where one does not necessarily need to assume that the relationship between them can be well approximated by a linear model.

A decision tree is a sequence of questions that can be answered as yes or no, plus a set of fitted response values. Each question asks whether a predictor satisfies a given condition. Predictors can be continuous or discrete. Depending on the answers to one question, one either proceeds to another question or arrives at a fitted response value.

A decision tree is built through a process known as binary recursive partitioning. This is an iterative process of splitting the data into partitions, and then splitting it up further on each of the branches. Initially all of the records in training set (used to determine the structure of the tree) are together. The

algorithm then tries breaking up the data, using every possible binary split on every field. The algorithm chooses the split that partitions the data into two parts such that it minimizes the sum of the squared deviations from the mean in the separate parts. This splitting or partitioning is then applied to each of the new branches. The process continues until each node reaches a user-specified minimum node size and becomes a terminal node. The regression trees give in this case on the "leaf" as final value a numerical value from 0 to 6. Classification tree analysis is a term used when the predicted outcome of the decision tree is the class to which the data belongs. Classification trees compared with other statistical methods are easily understood by both experts and non-experts and can provide a good illustration of the classification.

Another distinctive characteristic of classification and regression trees (CARTs - Brieman et al. 1984; Lee et al. 1997) is their flexibility. Their original ability is to examine the effects of the predictor variables one at a time, rather than all at once but there are several other ways in which they are more flexible than traditional regression analyses. The ability of classification and regression trees to perform univariate splits, examining the effects of predictors one at a time, has implications for the variety of types of predictors that can be analyzed. To summarize, CARTs can be computed for categorical predictors, continuous predictors, or any mix of the two types of predictors when univariate splits are used.

A critical issue on the CART´s design is obtaining right-sized trees, i.e. trees which neither underfit nor overfit the data is addressed. Instead of stopping rules to halt partitioning, the approach of growing a large tree with pure terminal nodes and selectively pruning it back is used.

The problems that can show up whenever working with CARTs is that the variables do not completely predict the outcome, or the data is noisy or incomplete (not all cases are covered).

Progress so far with Regression Trees

The idea behind the PARADISE model has already been used with regression and classification trees by Hastie et al. (2002a) to determine relevant predictors to user satisfaction. This approach covers only training data. The results

were similar to the ones from the linear regression model with an accuracy of R^2= 0.23…0.38 on training (in-sample) data. These results are limited as well, since they don't give rates on prediction of new users for instance. Experiments will be made using this same principle, but applying it on test (out-of-sample) data, using the 'leave-one-out' method, user by user.

Pruning technique towards generalization

Since the tree is based on the training (in-sample) data set, when it has reached the full structure it usually suffers from over-fitting, as if it is "explaining" random elements of the training (in-sample) data that are not likely to be features of the larger population of data. These results have poor performance on independent test (out-of-sample) data. Therefore, the regression tree must be pruned to achieve better generalization. This is correspondent to the "Early Stop Technique" from the neural networks approach. Several pruning techniques will be tried:

i. Pruning a percentage of the total amount of nodes.

ii. The usage of a 10-fold cross-validation, to find the 'best level' of pruning. The best level is the one that produces a regression tree that would provide in theory the best results on test (out-of-sample) data. The function from Matlab partitions the training sample into 10 subsamples, chosen randomly but with roughly equal size. For each subsample, the function fits a tree to the remaining data and uses it to predict the subsample. From this calculations a best level for the pruning is chosen.

iii. Using validation data to discover the best level of pruning.

iv. Pruning the amount of levels that would leave the tree with a fixed size (like pruning 'number of total levels -5' to leave a 5-level tree.

Two regression trees routines (meaning to different criteria for the 'split' of one node) will be tested as well with the help of MATLAB:

- 'catidx': values are treated as unordered categorical variables. The rules on the node can be either < and > or ∈ as a part of a group of categories together.

- 'splitmin': uses only < and > for the rule on each node, defining a minimum number of observations in order for that value to be considered.

In theory it should be used only 'splitmin' with a minimum number of observations set as one, since classifying the numerical variables of the interaction as categories would not make much sense (for example, the number of turns, 1,2,3 and 4 would be each one a category to be part of a group from a node rule, like part of [1 2 4 9]).

Known issues with classification and regression trees

i. Variables do not completely predict the outcome: meaning that more as well as better explanatory
variables are needed to make the prediction more effective.
ii. Data is noisy: meaning that adding some of the explanatory variables may actually not improve
the model in terms of prediction capability.
iii. Data is incomplete: meaning that not all cases are covered by the available data. More data is
necessary to build a complete prediction model using the regression trees.
iv. Difficult to scale: meaning that determining the "ideal" size of a tree in order not to overfit or
underfit the data is not an easy task. This is directly related to the generalization capabilities of the regression tree.

4.3
Model experimental settings

Four different tools were developed during the making of this work for the realization of the experiments. All the programs written for MATLAB are part of the CD attached to this dissertation. The programs have the objective to help conducting each experiment, in a quicker and more dynamic way. A pre-processing of the data is needed before using the programs, i.e. the selection of input variables needs to be done before using the MATLAB programs. A

Correlation test on MATLAB or SPSS as seen on chapter 4.1 is therefore needed. The post-processing is not needed, since the programs themselves present the results directly in the chosen statistic measures needed for comparison.

All programs use the leave-one-out method, to maximize the training data, since the amount of data existent is not the ideal for some of the methods (neural networks for instance).

From the dialogues from 40 users on BoRIS, the model would be trained with the dialogues from 39 users and would be tested with the dialogues from one independent user. This whole process would be then repeated 40 times so that at the end a vector with all the predictions can be filled and compared with the target values.

Since each model has its own particularities, they need to be enhanced and optimized in order to provide the best and most exact results. Several different configurations will be tried for each approach (for instance, different training functions or transfer functions on the neural networks approach) in order to optimize the results when it comes to predicting the target values and the most important particularities will be described on part III of this work and on the MATLAB programs themselves as commentaries.