diploma

1. Data set

1.1 Task description

The data set contains the information for creating the predictive model. It comprises a data matrix in which columns represent variables and rows represent instances. Variables in a data set can be of three types: The inputs will be the independent variables; the targets will be the dependent variables; the unused variables will neither be used as inputs nor as targets. Additionally, instances can be: Training instances, which are used to construct the model; selection instances, which are used for selecting the optimal order; testing instances, which are used to validate the functioning of the model; unused instances, which are not used at all.

1.2 Data preview table

The next table shows a preview of the data matrix contained in the file trainingData.csv. Here, the number of variables is 6, and the number of instances is 100.

	leftSensor	middleSensor	rightSensor	left	forward	right
1	60	35	70	0	1	0
2	53	76	15	1	0	0
...	...	...	...	...	...	...
100	39	64	51	0	1	0

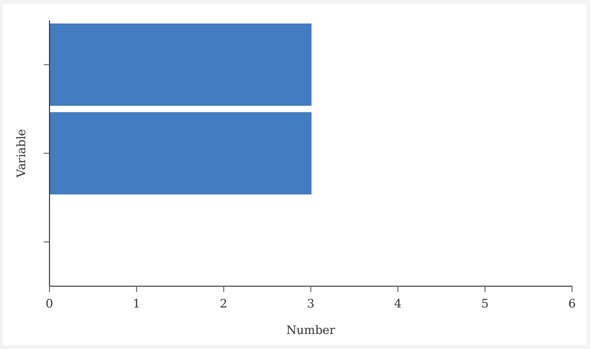
1.3 Variables table

The following table depicts the names, units, descriptions and uses of all the variables in the data set. The numbers of inputs, targets and unused variables here are 3, 3, and 0, respectively.

	Name	Units	Description	Use
1	leftSensor			Input
2	middleSensor			Input
3	rightSensor			Input
4	left			Target
5	forward			Target
6	right			Target

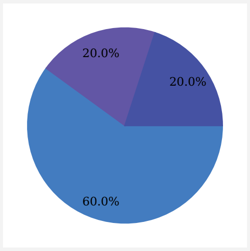
1.4 Variables bars chart

The next chart illustrates the variables use. It depicts the numbers of inputs (3), targets (3) and unused variables (0).



1.5 Instances pie chart

The following pie chart details the uses of all the instances in the data set. The total number of instances is 100. The number of training instances is 60 (60%), the number of selection instances is 20 (20%), the number of testing instances is 20 (20%), and the number of unused instances is 0 (0%).



1.6 Missing values results

There are not missing values in the data set.

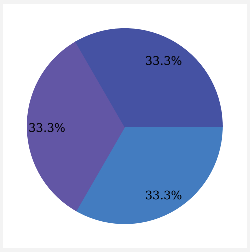
2. Target balancing

2.1 Task description

This task balances the distribution of target variables for a classification data set. It equals the number of instances of every target class by unusing those instances whose variables belong to the most populated bins. After this process, the distribution of the data will be more uniform and, in consequence, the model will probably be of better quality. Note that if the target distribution is very irregular then the number of instances to be unused could be big.

2.2 Target class distribution results

The number of unused instances has been set to 34. The next chart shows the number of instances belonging to each class in the data set. The number left, forward and right instances are 22, 22 and 22, respectively.



3. Training

3.1 Task description

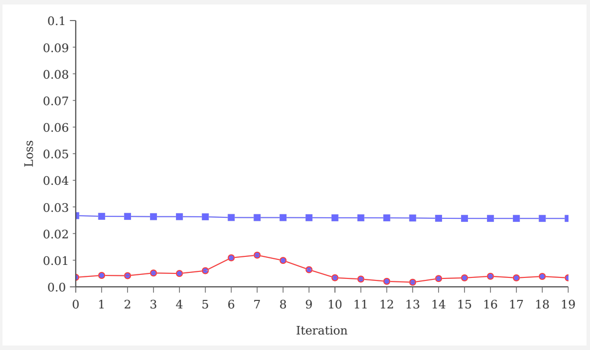
The procedure used to carry out the learning process is called training (or learning) strategy. The training strategy is applied to the neural network in order to obtain the best possible loss. The type of training is determined by the way in which the adjustment of the parameters in the neural network takes place.

3.2 Training algorithm

The quasi-Newton method is used here for training. It is based on Newton's method, but does not require calculation of second derivatives. Instead, the quasi-Newton method computes an approximation of the inverse Hessian at each iteration of the algorithm, by only using gradient information.

3.3 Quasi-Newton method losses history

The following plot shows the losses in each iteration. The initial value of the training loss is 0.0267354, and the final value after 19 iterations is 0.025681. The initial value of the selection loss is 0.00356601, and the final value after 19 iterations is 0.00337799.



3.4 Quasi-Newton method results

The next table shows the training results by the quasi-Newton method. They include some final states from the neural network, the loss functional and the training algorithm.

	Value
Final parameters norm	24.5
Final loss	0.0257
Final selection loss	0.00338
Final gradient norm	0.000989
Iterations number	19
Elapsed time	1
Stopping criterion	Gradient norm goal

4. Directional output

4.1 Task description

It is very useful to see the how the outputs vary as a function of a single input, when all the others are fixed. This can be seen as the cut of the neural network model along some input direction and through some reference point.

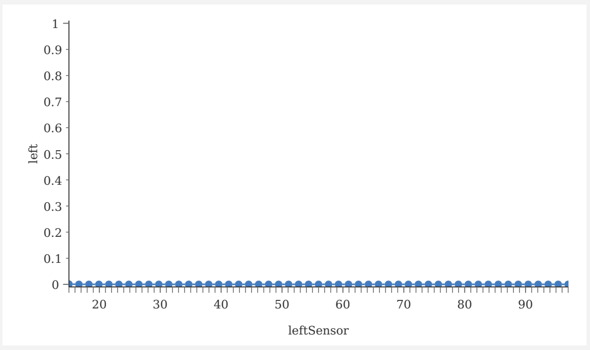
4.2 Reference point table

The next table shows the reference point for the plots.

	Value
leftSensor	49.7879
middleSensor	53.9394
rightSensor	53.9545

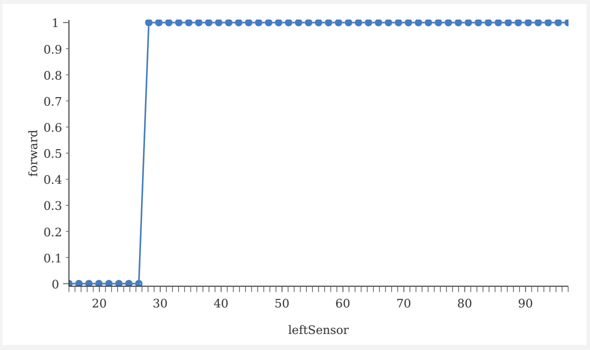
4.3 left against leftSensor directional line chart

The next plot shows the output left as a function of the input leftSensor. The x and y axes are defined by the range of the variables leftSensor and left, respectively.



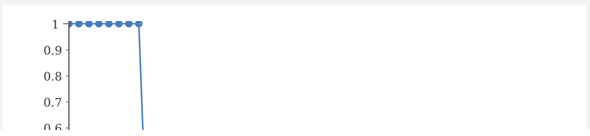
4.4 forward against leftSensor directional line chart

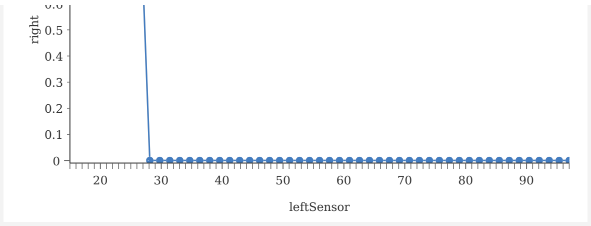
The next plot shows the output forward as a function of the input leftSensor. The x and y axes are defined by the range of the variables leftSensor and forward, respectively.



4.5 right against leftSensor directional line chart

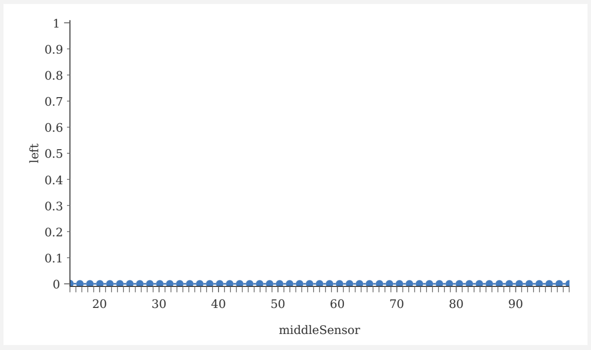
The next plot shows the output right as a function of the input leftSensor. The x and y axes are defined by the range of the variables leftSensor and right, respectively.





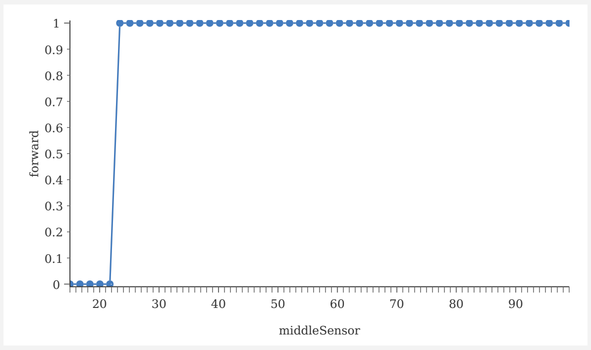
4.6 left against middleSensor directional line chart

The next plot shows the output left as a function of the input middleSensor. The x and y axes are defined by the range of the variables middleSensor and left, respectively.



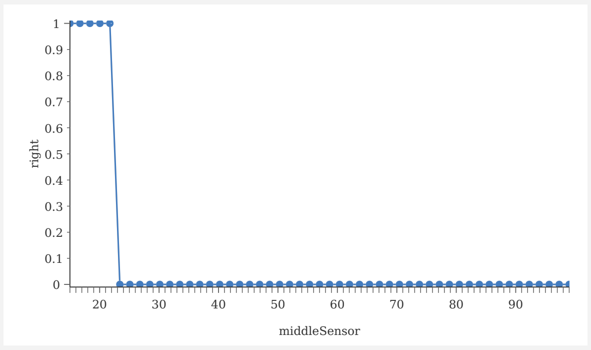
4.7 forward against middleSensor directional line chart

The next plot shows the output forward as a function of the input middleSensor. The x and y axes are defined by the range of the variables middleSensor and forward, respectively.



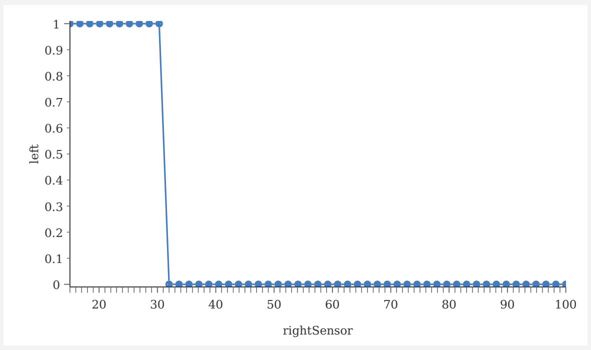
4.8 right against middleSensor directional line chart

The next plot shows the output right as a function of the input middleSensor. The x and y axes are defined by the range of the variables middleSensor and right, respectively.



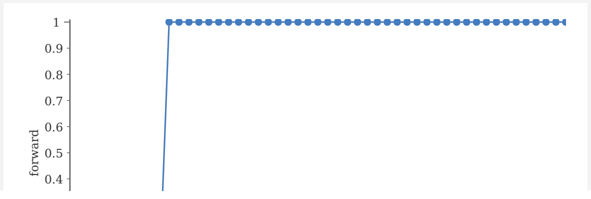
4.9 left against rightSensor directional line chart

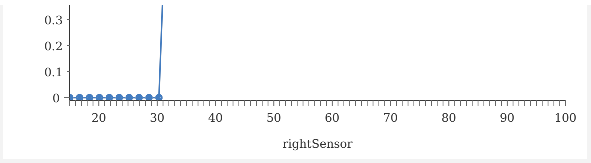
The next plot shows the output left as a function of the input rightSensor. The x and y axes are defined by the range of the variables rightSensor and left, respectively.



4.10 forward against rightSensor directional line chart

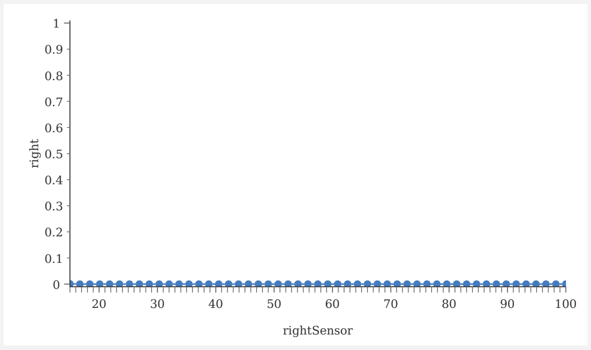
The next plot shows the output forward as a function of the input rightSensor. The x and y axes are defined by the range of the variables rightSensor and forward, respectively.





4.11 right against rightSensor directional line chart

The next plot shows the output right as a function of the input rightSensor. The x and y axes are defined by the range of the variables rightSensor and right, respectively.



5. Python expression

5.1 Task description

The predictive model takes the form of a function of the outputs with respect to the inputs. The mathematical expression represented by the model can be exported to different programming languages, in the so called production mode. The Python programming language expression has been saved in the following file: /development/master.thesis/model.py

6. Correlation matrix

6.1 Task description

This task calculates the absolute values of the linear correlations among all inputs. The correlation is a numerical value between 0 and 1 that expresses the strength of the relationship between two variables. When it is close to 1 it indicates a strong relationship, and a value close to 0 indicates that there is no relationship.

6.2 Correlation matrix

The following table shows the absolute value of the correlations between all input variables. The minimal correlation is 0.0266638 between the variables middleSensor and rightSensor. The maximal correlation is 0.236989 between the variables leftSensor and middleSensor.

	leftSensor	middleSensor	rightSensor
leftSensor	1	0.237	0.191
middleSensor		1	0.0267
rightSensor			1

7. Logistic correlations

7.1 Task description

In classification applications, it might be interesting to look for logistic dependencies between single input and single target variables. This task calculates the absolute values of the logistic correlation between all inputs and all targets. The logistic correlation is a numerical value between 0 and 1 that expresses the strength of the logistic relationship between a single input and output variables. When it is close to 1 it indicates a strong relationship. A value close to 0 indicates that there is no relationship.

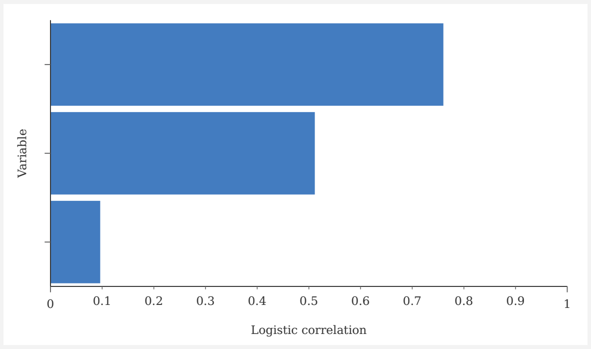
7.2 Logistic correlations

The following table shows the absolute value of the logistic correlations between all input and target variables. The maximum correlation (0.886938) is yield between the input variable leftSensor and the target variable right.

	left	forward	right
leftSensor	0.51	0.41	0.887
middleSensor	0.0948	0.334	0.0449
rightSensor	0.759	0.235	0.38

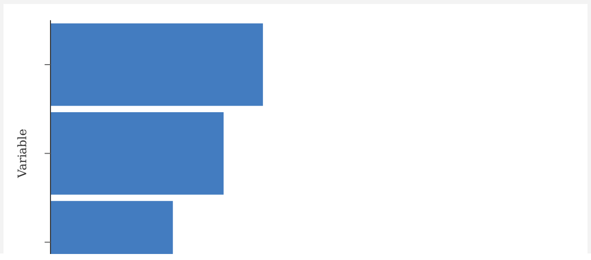
7.3 left bars chart

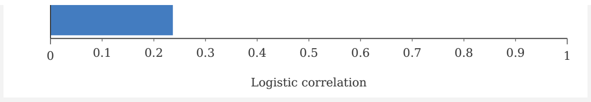
The next chart illustrates the dependency of the target left with all the input variables.



7.4 forward bars chart

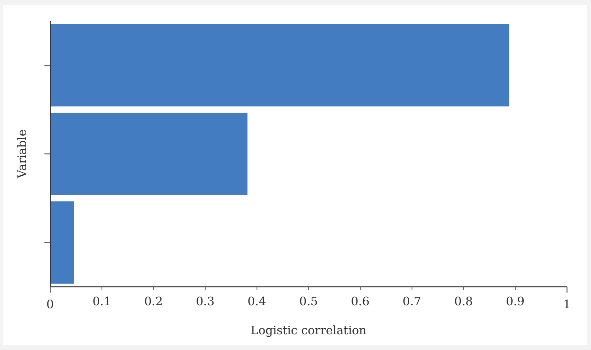
The next chart illustrates the dependency of the target forward with all the input variables.





7.5 right bars chart

The next chart illustrates the dependency of the target right with all the input variables.



8. Neural network

8.1 Task description

The neural networks represents the predictive model. In Neural Designer neural networks allow deep architectures, which are a class of universal approximator.

8.2 Inputs

The number of inputs is 3. The next table depicts some basic information about them, including the name, the units and the description.

	Name	Units	Description
1	leftSensor		
2	middleSensor		
3	rightSensor		

8.3 Scaling layer

The size of the scaling layer is 3, the number of inputs. The scaling method for this layer is the MinimumMaximum. The following table shows the values which are used for scaling the inputs, which include the minimum, maximum, mean and standard deviation.

	Minimum	Maximum	Mean	Deviation
leftSensor	15	97	49.8	26.6
middleSensor	15	99	53.9	27.1
rightSensor	15	100	54	26.8

8.4 Neural network

The number of layers in the neural network is 3. The following table depicts the size of each layer and its corresponding activation function. The architecture of this neural network can be written as 3:5:3:3.

	Inputs number	Neurons number	Activation function
1	3	5	HyperbolicTangent
2	5	3	Logistic
3	3	3	Logistic

8.5 Neural network parameters

The following table shows the statistics of the parameters of the neural network. The total number of parameters is 50.

	Minimum	Maximum	Mean	Standard deviation
Statistics	-8.26	8.84	-0.43	3.48

8.6 Probabilistic layer

The size of the probabilistic layer is 3, the number of outputs. The probabilistic method for this layer is the competitive.

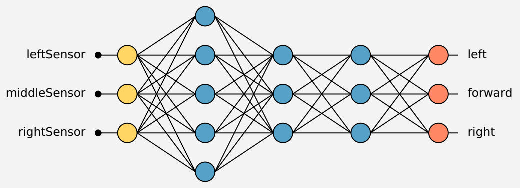
8.7 Outputs table

The number of outputs is 3. The next table depicts some basic information about them, including the name, the units and the description.

	Name	Units	Description
1	left		
2	forward		
3	right		

8.8 Neural network graph

A graphical representation of the network architecture is depicted next. It contains a scaling layer, a neural network and a probabilistic layer. The yellow circles represent scaling neurons, the blue circles perceptron neurons and the red circles probabilistic neurons. The number of inputs is 3, and the number of outputs is 3. The complexity, represented by the numbers of hidden neurons, is 5:3.



9. Parameters norm

9.1 Task description

The norm of the parameters gives a clue about the complexity of the predictive model. If the parameters norm is small, the model will be smooth. On the other hand, if the parameters norm is very big, the model might become unstable. Also note that the norm depends on the number of parameters.

9.2 Parameters norm results

The norm of the neural parameters is written below. The architecture of this neural network is 3:5:3:3, and the number of parameters is 50.

	Value
Parameters norm	24.5

10. Parameters statistics

10.1 Task description

The statistics of the parameters depict information about the complexity of the model. In general, it is desirable that all the minimum, maximum, mean and standard deviation values are not very big.

10.2 Parameters statistics table

The table below shows the minimum, maximum, mean and standard deviation of the parameters in the neural network.

	Minimum	Maximum	Mean	Deviation
Parameters	-8.25648	8.83799	-0.430214	3.47992

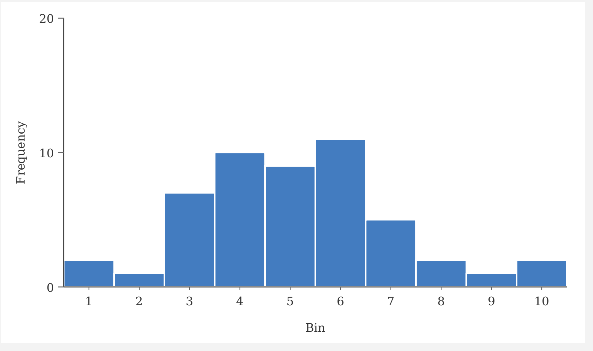
11. Parameters histogram

11.1 Description

The histogram of the parameters shows how they are distributed. A regular distribution for the parameters is, in general, desirable. If the parameters are very irregularly distributed, then the model is probably unstable.

11.2 Parameters histogram chart

The following chart shows the histogram for the parameters. The abscissa represents the centers of the containers, and the ordinate their corresponding frequencies. The minimum frequency is 1, which corresponds to the bins with centers -5.69231 and 6.27382. The maximum frequency is 11, which corresponds to the bin with center 1.14548.



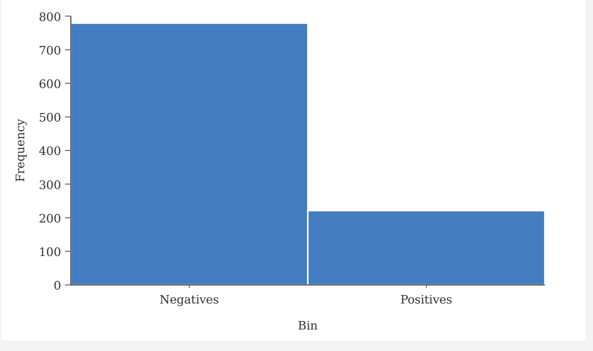
12. Outputs histogram

12.1 Description

The histogram of the outputs shows how they are distributed. This method takes 1000 random instances and calculate the histogram with its outputs.

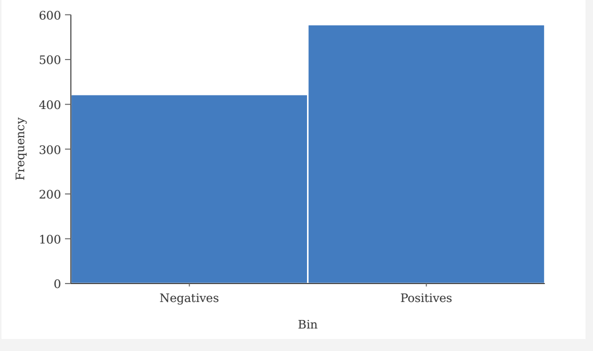
12.2 left histogram

The following chart shows the histogram for the output left. The abscissa represents the centers of the containers, and the ordinate their corresponding frequencies. The minimum frequency is 221, which corresponds to the bin with center 1. The maximum frequency is 779, which corresponds to the bin with center 0.



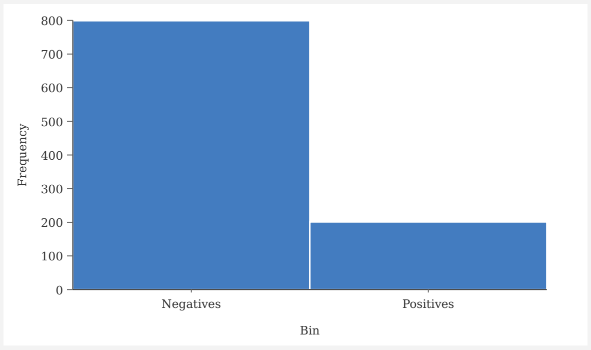
12.3 forward histogram

The following chart shows the histogram for the output forward. The abscissa represents the centers of the containers, and the ordinate their corresponding frequencies. The minimum frequency is 422, which corresponds to the bin with center 0. The maximum frequency is 578, which corresponds to the bin with center 1.



12.4 right histogram

The following chart shows the histogram for the output right. The abscissa represents the centers of the containers, and the ordinate their corresponding frequencies. The minimum frequency is 201, which corresponds to the bin with center 1. The maximum frequency is 799, which corresponds to the bin with center 0.



13. Loss index

13.1 Task description

The loss index plays an important role in the use of a neural network. It defines the task the neural network is required to do, and provides a measure of the quality of the representation that it is required to learn. The choice of a suitable loss index depends on the particular application.

13.2 Error method

The normalized squared error is used here as the error method. It divides the squared error between the outputs from the neural network and the targets in the data set by a normalization coefficient. If the normalized squared error has a value of unity then the neural network is predicting the data 'in the mean', while a value of zero means perfect prediction of the data.

13.3 Regularization method

The neural parameters norm is used as the regularization method. Is is applied to control the complexity of the neural network by reducing the value of the parameters. The following table shows the weight of this regularization term in the loss expression.

	Value
Neural parameters norm weight	0.001

14. Training strategy

14.1 Task description

The procedure used to carry out the learning process is called training (or learning) strategy. The training strategy is applied to the neural network in order to obtain the best possible loss.

14.2 Training algorithm

The quasi-Newton method is used here as training algorithm. It is based on Newton's method, but does not require calculation of second derivatives. Instead, the quasi-Newton method computes an approximation of the inverse Hessian at each iteration of the algorithm, by only using gradient information.

	Description	Value
Inverse Hessian approximation method	Method used to obtain a suitable training rate.	BFGS
Training rate method	Method used to calculate the step for the quasi-Newton training direction.	BrentMethod
Training rate tolerance	Maximum interval length for the training rate.	0.005
Minimum parameters increment norm	Norm of the parameters increment vector at which training stops.	1e-09
Minimum loss increase	Minimum loss improvement between two successive iterations.	1e-12
Performance goal	Goal value for the loss.	1e-12
Gradient norm goal	Goal value for the norm of the objective function gradient.	0.001
Maximum selection loss increases	Maximum number of iterations at which the selection loss increases.	100
Maximum iterations number	Maximum number of iterations to perform the training.	1000
Maximum time	Maximum training time.	3600
Reserve parameters norm history	Plot a graph with the parameters norm of each iteration.	false
Reserve loss history	Plot a graph with the loss of each iteration.	true
Reserve selection loss history	Plot a graph with the selection loss of each iteration.	true
Reserve gradient norm history	Plot a graph with the gradient norm of each iteration.	false

15. Model selection

15.1 Task description

Model selection is applied to find a neural network with a topology that optimizes the loss on new data. There are two different types of algorithms for model selection: Order selection algorithms and input selection algorithms. Order selection algorithms are used to find the optimal number of hidden neurons in the network. Inputs selection algorithms are responsible for finding the optimal subset of input variables.

15.2 Inputs selection algorithm

The inputs selection algorithm chosen for this application is growing inputs. With this method, the inputs are added progressively based on their correlations with the targets.

	Description	Value
Trials number	Number of trials for each neural network.	3
Tolerance	Tolerance for the selection loss in the trainings of the algorithm.	0.01
Selection loss goal	Goal value for the selection loss.	0
Maximum selection failures	Maximum number of iterations at which the selection loss increases.	3
Maximum inputs number	Maximum number of inputs in the neural network.	3
Minimum correlation	Minimum value for the correlations to be considered.	0
Maximum correlation	Maximum value for the correlations to be considered.	1
Maximum iterations number	Maximum number of iterations to perform the algorithm.	100
Maximum time	Maximum time for the inputs selection algorithm.	3600
Plot training loss history	Plot a graph with the training losses of each iteration.	true
Plot selection loss history	Plot a graph with the selection losses of each	true

15.3 Order selection algorithm

The order selection algorithm chosen for this application is incremental order. This method start with the minimum order and adds a given number of perceptrons in each iteration.

	Description	Value
Minimum order	Number of minimum hidden perceptrons to be evaluated.	1
Maximum order	Number of maximum hidden perceptrons to be evaluated.	10
	Number of hidden perceptrons added in each	

Step	Number of hidden perceptrons added in each iteration.	1
Trials number	Number of trials for each neural network.	3
Tolerance	Tolerance for the selection loss in the trainings of the algorithm.	0.01
Selection loss goal	Goal value for the selection loss.	0
Maximum selection failures	Maximum number of iterations at which the selection loss increases.	5
Maximum iterations number	Maximum number of iterations to perform the algorithm.	1000
Maximum time	Maximum time for the order selection algorithm.	3600
Plot training loss history	Plot a graph with the training losses of each iteration.	true
Plot selection loss history	Plot a graph with the selection losses of each iteration.	true

16. Inputs importance

16.1 Task description

This task calculates the selection loss when removing one input at a time. This shows which input have more influence in the outputs.

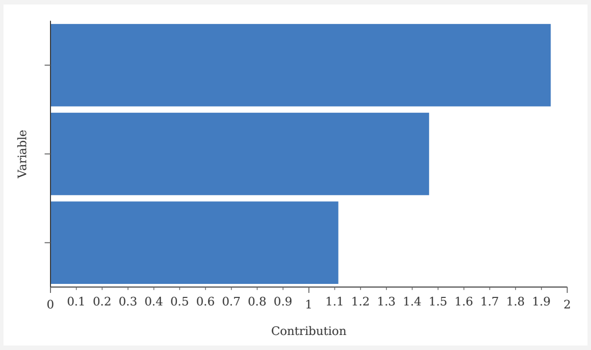
16.2 Inputs importance results

The next table shows the importance of each input. If the importance takes a value greater than 1 for an input, it means that the selection error without that input is greater than with it. In the case that the importance is lower than 1, the selection error is lower without using that input. Finally, if the importance is 1, there is no difference between using the current input and not using it. The most important variable is leftSensor, that gets a contribution of 193.3% to the outputs.

	Contribution
leftSensor	1.93
middleSensor	1.46
rightSensor	1.11

16.3 Contribution bars chart

The next chart illustrates the contribution of each input.



17. Inputs selection

17.1 Task description

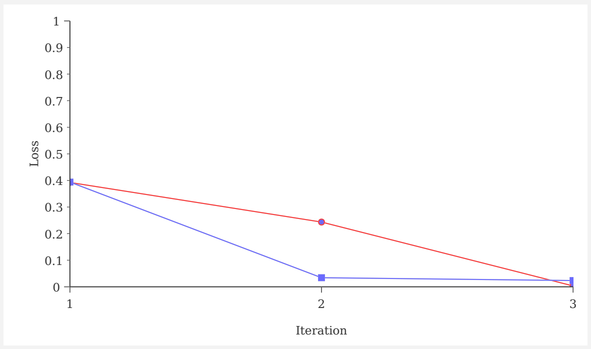
Some data sets have inputs that are redundants and it affects the loss of the neural network. The inputs selection are used to find the optimal subset of inputs for the best loss of the model.

17.2 Inputs selection algorithm

Growing inputs is used here as inputs selection algorithm in the model selection.

17.3 Growing inputs losses plot

The next chart shows the loss history for the different subsets during the growing inputs selection process. The blue line represents the training loss and the red line symbolizes the selection loss.



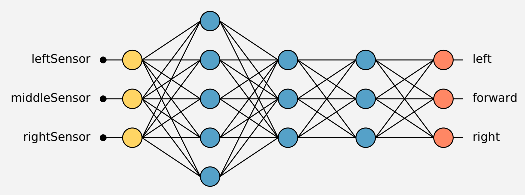
17.4 Growing inputs results

The next table shows the inputs selection results by the growing inputs algorithm. They include some final states from the neural network, the loss functional and the inputs selection algorithm.

	Value
Optimal number of inputs	3
Optimum training loss	0.0235556
Optimum selection loss	0.0033118 1
Iterations number	3
Elapsed time	2

17.5 Final architecture

A graphical representation of the resulted deep architecture is depicted next. It contains a scaling layer, a neural network and a probabilistic layer. The yellow circles represent scaling neurons, the blue circles perceptron neurons and the red circles probabilistic neurons. The number of inputs is 3, and the number of outputs is 3. The complexity, represented by the numbers of hidden neurons, is 5:3.



18. Order selection

18.1 Task description

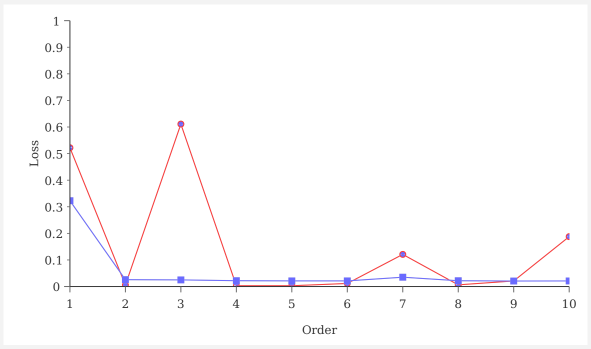
The best selection is achieved by using a model whose complexity is the most appropriate to produce an adequate fit of the data. The order selection algorithm is responsible of finding the optimal number of neurons in the network.

18.2 Order selection algorithm

Incremental order is used here as order selection algorithm in the model selection.

18.3 Incremental order losses plot

The next chart shows the loss history for the different subsets during the incremental order selection process. The blue line represents the training loss and the red line symbolizes the selection loss.



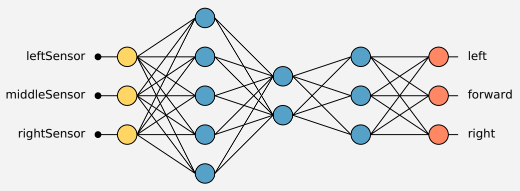
18.4 Incremental order results

The next table shows the order selection results by the incremental order algorithm. They include some final states from the neural network, the loss functional and the order selection algorithm.

	Value
Optimal order	2
Optimum training loss	0.0258428
Optimum selection loss	0.0056716 4
Iterations number	10
Elapsed time	4

18.5 Final architecture

A graphical representation of the resulted deep architecture is depicted next. It contains a scaling layer, a neural network and a probabilistic layer. The yellow circles represent scaling neurons, the blue circles perceptron neurons and the red circles probabilistic neurons. The number of inputs is 3, and the number of outputs is 3. The complexity, represented by the numbers of hidden neurons, is 5:2.



19. Neural network

19.1 Task description

The neural networks represents the predictive model. In Neural Designer neural networks allow deep architectures, which are a class of universal approximator.

19.2 Inputs

The number of inputs is 3. The next table depicts some basic information about them, including the name, the units and the description.

	Name	Units	Description
1	leftSensor		
2	middleSensor		
3	rightSensor		

19.3 Scaling layer

The size of the scaling layer is 3, the number of inputs. The scaling method for this layer is the MinimumMaximum. The following table shows the values which are used for scaling the inputs, which include the minimum, maximum, mean and standard deviation.

	Minimum	Maximum	Mean	Deviation
leftSensor	15	97	49.8	26.6
middleSensor	15	99	53.9	27.1
rightSensor	15	100	54	26.8

19.4 Neural network

The number of layers in the neural network is 3. The following table depicts the size of each layer and its corresponding activation function. The architecture of this neural network can be written as 3:5:2:3.

	Inputs number	Neurons number	Activation function
1	3	5	HyperbolicTangent
2	5	2	Logistic
3	2	3	Logistic

19.5 Neural network parameters

The following table shows the statistics of the parameters of the neural network. The total number of parameters is 41.

	Minimum	Maximum	Mean	Standard deviation
Statistics	-8.23	8.11	-0.261	3.71

19.6 Probabilistic layer

The size of the probabilistic layer is 3, the number of outputs. The probabilistic method for this layer is the competitive.

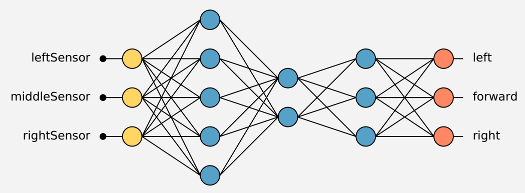
19.7 Outputs table

The number of outputs is 3. The next table depicts some basic information about them, including the name, the units and the description.

	Name	Units	Description
1	left		
2	forward		
3	right		

19.8 Neural network graph

A graphical representation of the network architecture is depicted next. It contains a scaling layer, a neural network and a probabilistic layer. The yellow circles represent scaling neurons, the blue circles perceptron neurons and the red circles probabilistic neurons. The number of inputs is 3, and the number of outputs is 3. The complexity, represented by the numbers of hidden neurons, is 5:2.



20. Parameters norm

20.1 Task description

The norm of the parameters gives a clue about the complexity of the predictive model. If the parameters norm is small, the model will be smooth. On the other hand, if the parameters norm is very big, the model might become unstable. Also note that the norm depends on the number of parameters.

20.2 Parameters norm results

The norm of the neural parameters is written below. The architecture of this neural network is 3:5:2:3, and the number of parameters is 41.

	Value
Parameters norm	23.5

21. Parameters norm

21.1 Task description

The norm of the parameters gives a clue about the complexity of the predictive model. If the parameters norm is small, the model will be smooth. On the other hand, if the parameters norm is very big, the model might become unstable. Also note that the norm depends on the number of parameters.

21.2 Parameters norm results

The norm of the neural parameters is written below. The architecture of this neural network is 3:5:2:3, and the number of parameters is 41.

	Value
Parameters norm	23.5

22. Parameters statistics

22.1 Task description

The statistics of the parameters depict information about the complexity of the model. In general, it is desirable that all the minimum, maximum, mean and standard deviation values are not very big.

22.2 Parameters statistics table

The table below shows the minimum, maximum, mean and standard deviation of the parameters in the neural network.

	Minimum	Maximum	Mean	Deviation
Parameters	-8.23083	8.11208	-0.260997	3.70693

23. Loss index

23.1 Task description

The loss index plays an important role in the use of a neural network. It defines the task the neural network is required to do, and provides a measure of the quality of the representation that it is required to learn. The choice of a suitable loss index depends on the particular application.

23.2 Error method

The normalized squared error is used here as the error method. It divides the squared error between the outputs from the neural network and the targets in the data set by a normalization coefficient. If the normalized squared error has a value of unity then the neural network is predicting the data 'in the mean', while a value of zero means perfect prediction of the data.

23.3 Regularization method

The neural parameters norm is used as the regularization method. Is is applied to control the complexity of the neural network by reducing the value of the parameters. The following table shows the weight of this regularization term in the loss expression.

	Value
Neural parameters norm weight	0.001

24. Training strategy

24.1 Task description

The procedure used to carry out the learning process is called training (or learning) strategy. The training strategy is applied to the neural network in order to obtain the best possible loss.

24.2 Training algorithm

The quasi-Newton method is used here as training algorithm. It is based on Newton's method, but does not require calculation of second derivatives. Instead, the quasi-Newton method computes an approximation of the inverse Hessian at each iteration of the algorithm, by only using gradient information.

	Description	Value
Inverse Hessian approximation method	Method used to obtain a suitable training rate.	BFGS
Training rate method	Method used to calculate the step for the quasi-Newton training direction.	BrentMethod
Training rate tolerance	Maximum interval length for the training rate.	0.005
Minimum parameters increment norm	Norm of the parameters increment vector at which training stops.	1e-09
Minimum loss increase	Minimum loss improvement between two successive iterations.	1e-12
Performance goal	Goal value for the loss.	1e-12
Gradient norm goal	Goal value for the norm of the objective function gradient.	0.001
Maximum selection loss increases	Maximum number of iterations at which the selection loss increases.	100
Maximum iterations number	Maximum number of iterations to perform the training.	1000
Maximum time	Maximum training time.	3600
Reserve parameters norm history	Plot a graph with the parameters norm of each iteration.	false
Reserve loss history	Plot a graph with the loss of each iteration.	true

Reserve loss history	Plot a graph with the loss of each iteration.	true
Reserve selection loss history	Plot a graph with the selection loss of each iteration.	true
Reserve gradient norm history	Plot a graph with the gradient norm of each iteration.	false

25. Training

25.1 Task description

The procedure used to carry out the learning process is called training (or learning) strategy. The training strategy is applied to the neural network in order to obtain the best possible loss. The type of training is determined by the way in which the adjustment of the parameters in the neural network takes place.

25.2 Training algorithm

The quasi-Newton method is used here for training. It is based on Newton's method, but does not require calculation of second derivatives. Instead, the quasi-Newton method computes an approximation of the inverse Hessian at each iteration of the algorithm, by only using gradient information.

25.3 Quasi-Newton method results

The next table shows the training results by the quasi-Newton method. They include some final states from the neural network, the loss functional and the training algorithm. Note that the number of iterations needed to converge is zero, which means that the training algorithm did not modify the state of the neural network.

	Value
Final parameters norm	23.5
Final loss	0.0258
Final selection loss	0.00565
Final gradient norm	0.000959
Iterations number	0
Elapsed time	0
Stopping criterion	Gradient norm goal

26. Python expression

26.1 Task description

The predictive model takes the form of a function of the outputs with respect to the inputs. The mathematical expression represented by the model can be exported to different programming languages, in the so called production mode. The Python programming language expression has been saved in the following file: /development/master.thesis/model.py