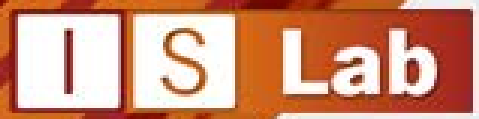
Natural Computing

Artificial Neural Networks (II)

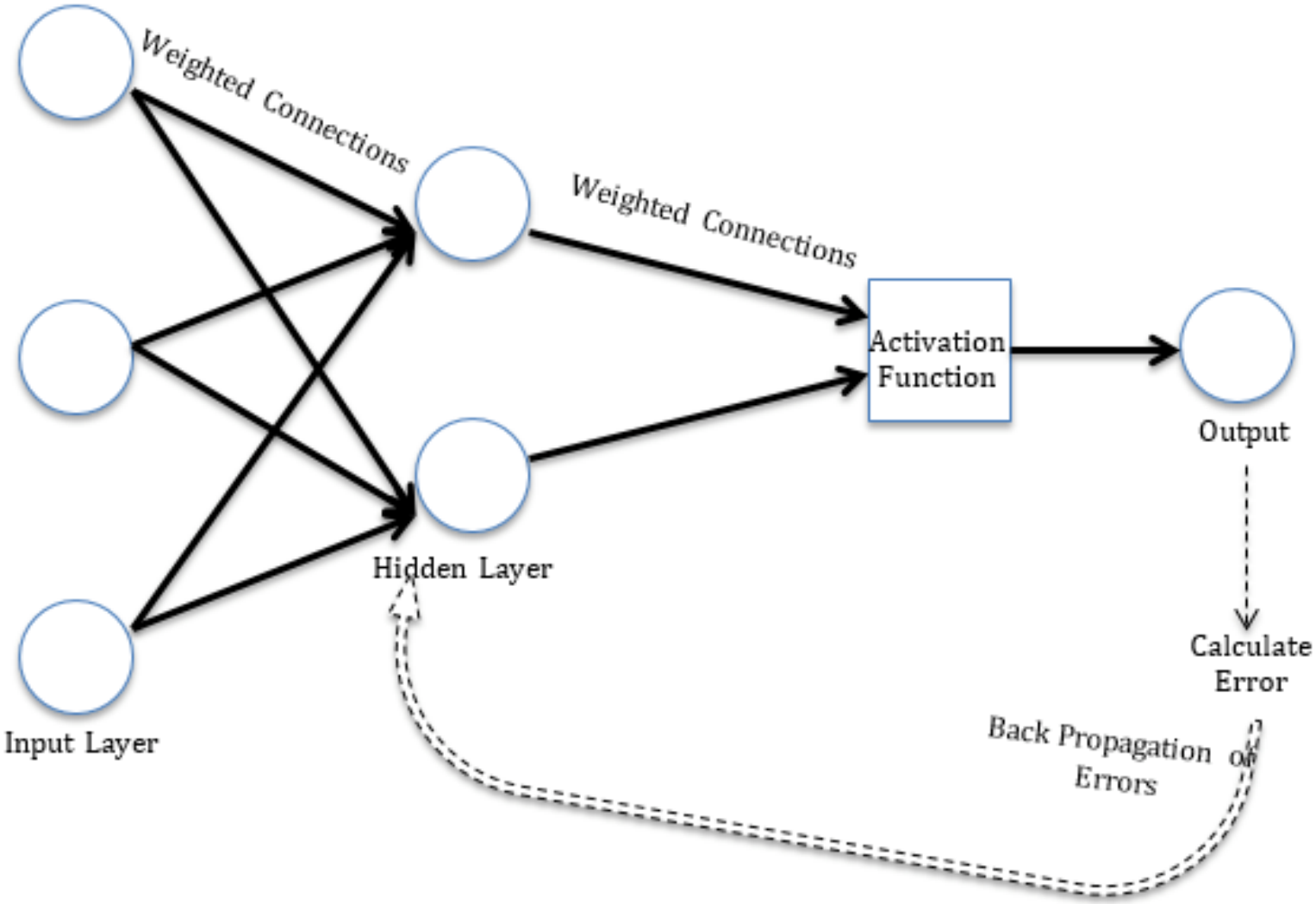


Intelligent Systems Lab

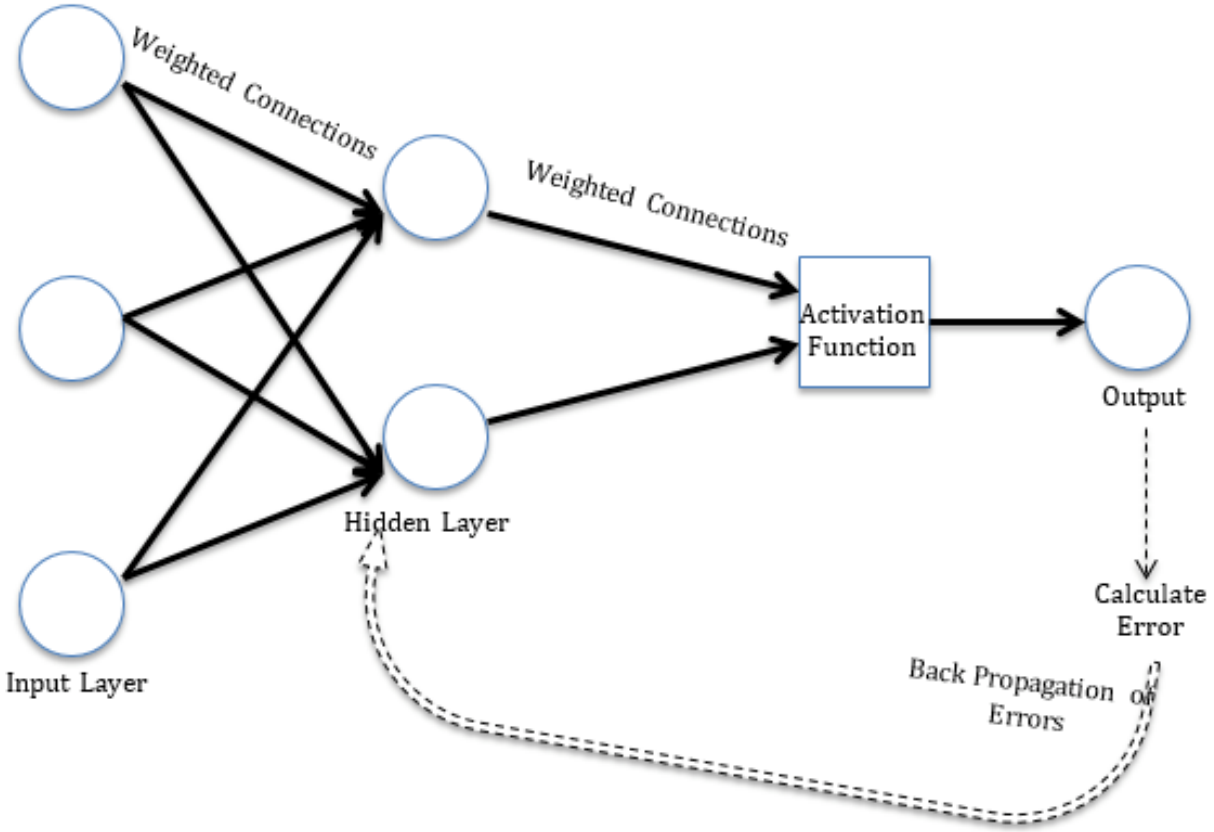
Universidade do Minho

João Ramos (jramos@di.uminho.pt)
André Pimenta (apimenta@di.uminho.pt)
Paulo Novais (pjon@di.uminho.pt)

Neural Network



Neural Network



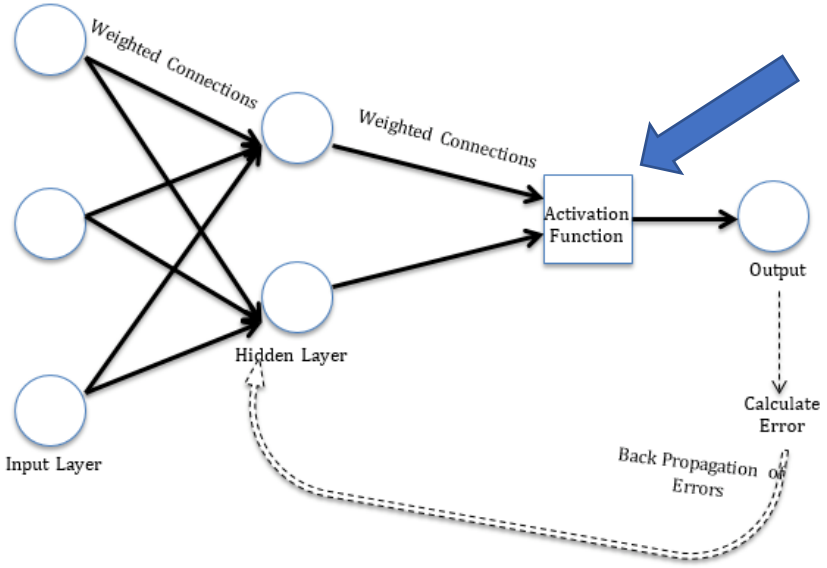
There are many learning rules that are used with neural network:

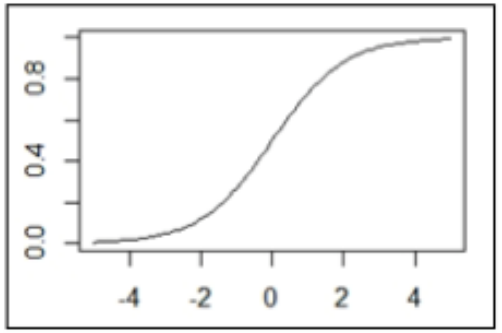
- least mean square;
- gradient descent;
- newton's rule;
- conjugate gradient etc.

The learning rules can be used in conjunction with backpropgation error method.

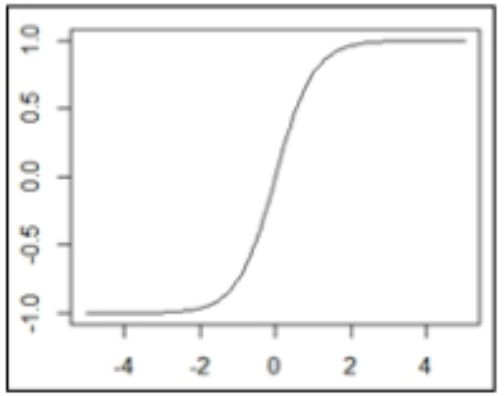
The learning rule is used to calculate the error at the output unit. This error is backpropagated to all the units such that the error at each unit is proportional to the contribution of that unit towards total error at the output unit. The errors at each unit are then used to optimize the weight at each connection.

Neural Network



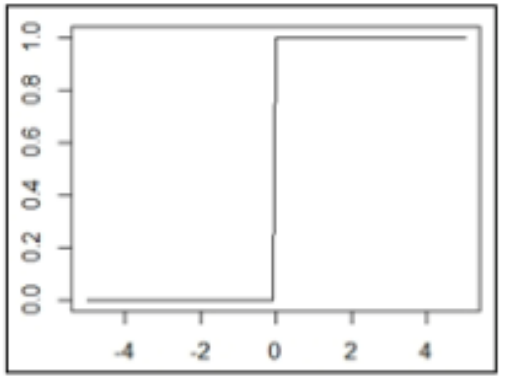


Sigmoid

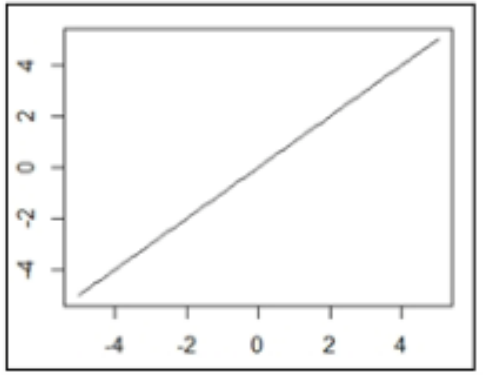


Hyperbolic tangent

Hard limiting threshold



Linear



Neural Network

- Create dataset
- Separate into training and testing sets
- Scale the sets
- Create and train the NN
  - Define the formula
  - Set the hidden layers
  - Set learning rate interval
  - Set the max number of step
  - Set the number of repetitions
- Check the NN with the testing set
- Calculate the error

Bank DataSet

	Vencim/o Mensal	Habitação	Automóvel	Cartão de Crédito	Perc. do Vencim/o	Perc. do Total	Avaliação
1	10 000	2 500	600	300	25,0%	34,0%	Sim
2	9 000	4 500	900	900	50,0%	70,0%	Não
3	8 000	2 000	900	1 100	25,0%	50,0%	Não
4	7 000	2 000	650	200	28,6%	40,7%	Não
5	6 000	1 500	300	200	25,0%	33,3%	Sim
6	5 000	2 700	900	780	54,0%	87,6%	Não
7	4 500	1 100	300	100	24,4%	33,3%	Sim
8	4 000	1 200	400	0	30,0%	40,0%	Não
9	3 500	800	300	100	22,9%	34,3%	Sim
10	3 000	600	200	100	20,0%	30,0%	Sim
11	2 500	500	250	50	20,0%	32,0%	Sim
12	2 000	380	150	0	19,0%	26,5%	Sim
13	1 500	600	250	100	40,0%	63,3%	Não
14	1 000	270	100	50	27,0%	42,0%	Não
15	500	200	250	0	40,0%	90,0%	Não

Bank DataSet

- Change directory:
  - setwd()
- Read data:
  - data <- read.delim(file_name,header = TRUE, sep=" ",dec = ".")
- Normalize the data
  - max <- apply(data,2,max)
  - min <- apply(data,2,min)
  - data <- as.data.frame(scale(data,center = min, scale = max-min))
- Create test and train sets
  - sample.split(data$Vencimento, SplitRatio = 0.70)
  - train = subset(data, split == TRUE)
  - test = subset(data, split == FALSE)

Bank DataSet

- Create the formula
  - Avaliacao ~ Vencimento + Habitacao + Automovel + Cartao
- Train the NN
  - nn <- neuralnet(f,train,hidden=3,linear.output=TRUE)
- Test the NN
  - compute(nn, test[1:4])
- Re-scale the values
  - predicted.nn.values$net.result*(max[5]-min[5])+min[5]
- Calculate the error
  - MSE – Mean square error
  - RMSE – Root mean square error

Bank DataSet – Using the normalized set

- Change directory:
  - setwd()
- Read data:
  - data <- read.delim(file_name,header = TRUE, sep=" ",dec = ".")
- Create test and train sets
  - sample.split(data$Vencimento, SplitRatio = 0.70)
  - train = subset(data, split == TRUE)
  - test = subset(data, split == FALSE)
- Create the formula
  - Avaliacao ~ Vencimento + Habitacao + Automovel + Cartao
- Train the NN
  - nn <- neuralnet(f,train,hidden=3,linear.output=TRUE)

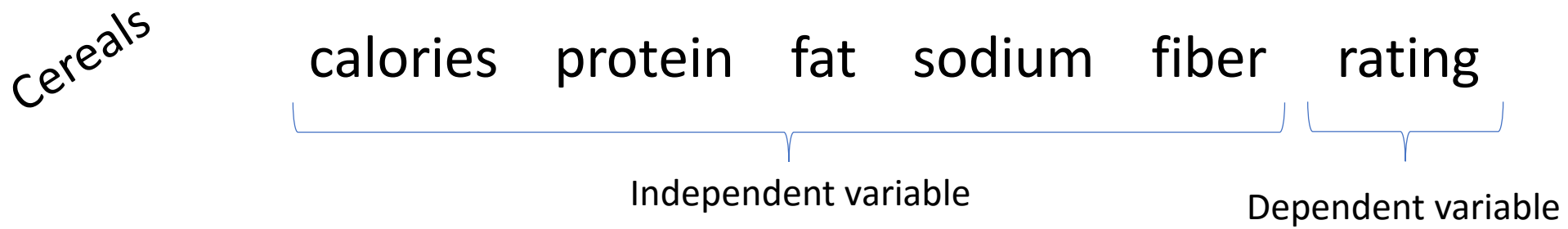
Bank DataSet – Using the normalized set

- Test the NN
  - compute(nn, test[1:4])
- Calculate the error
  - MSE – Mean square error
  - RMSE – Root mean square error

Bank DataSet – Using the normalized set

- Read the new test cases
- Compute these values with the NN
- Optional (convert to 0/1 values):
  - sapply(predicted.nn.values$net.result,round,digits=0)

Fitting Neural Network in R

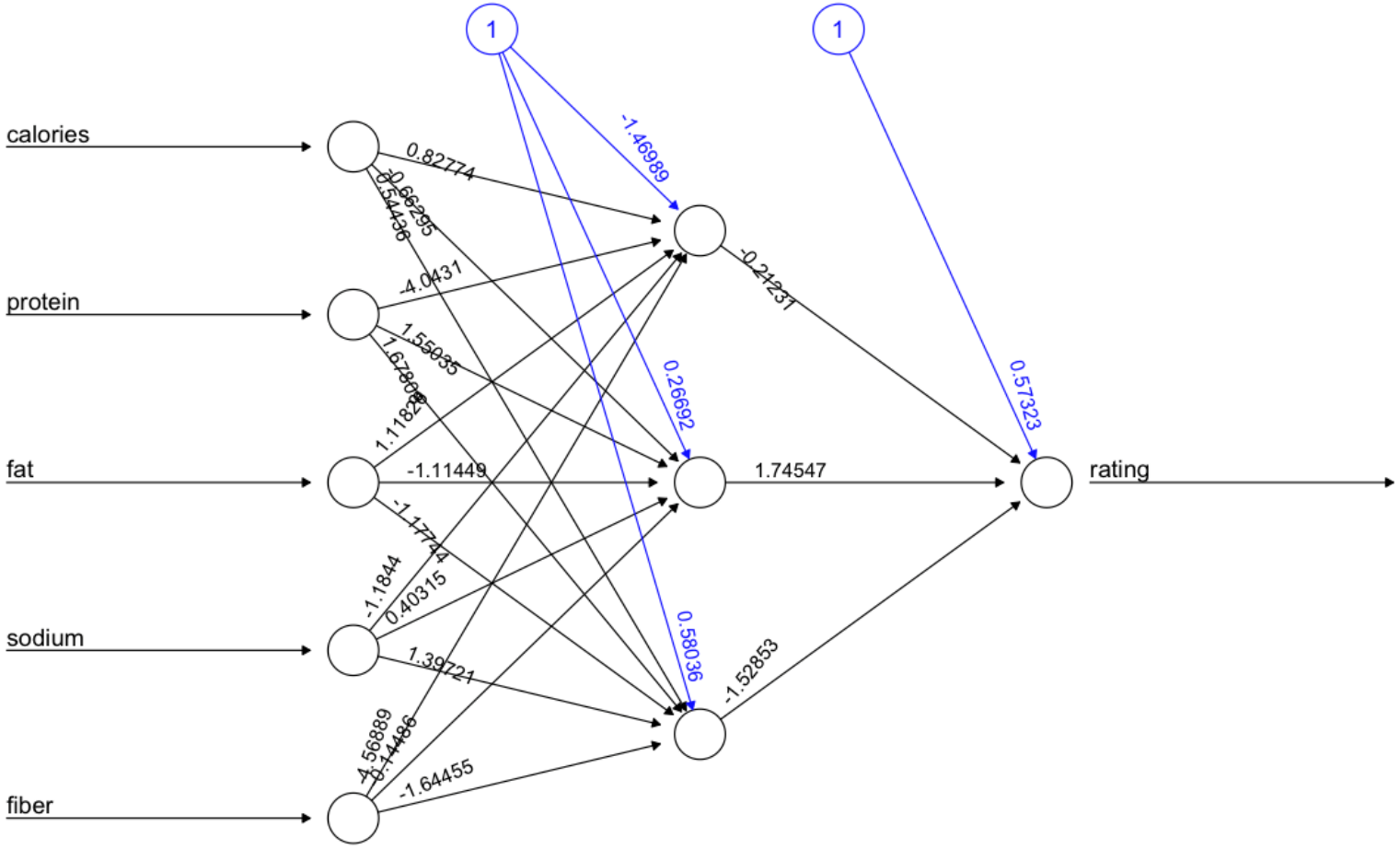


- Training set is used to find the relationship between dependent and independent variables (60% of the set)
- The test set assesses the performance of the model

- Normalization: The min-max normalization transforms the data into a common range, thus removing the scaling effect from all the variables

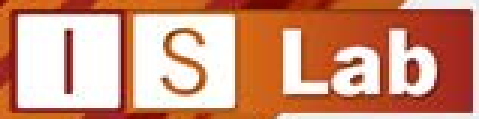
Fitting Neural Network in R

- 3 hidden layers
- seed(80)



Natural Computing

Artificial Neural Networks (II)



Intelligent Systems Lab

Universidade do Minho

João Ramos (jramos@di.uminho.pt)
André Pimenta (apimenta@di.uminho.pt)
Paulo Novais (pjon@di.uminho.pt)