sensitivi_met_data

Ana Costa Conrado

October 19, 2018

Neural network with neuralnet

```
install.packages(c("neuralnet","GGally", "tidyverse","car"), repos= "http://cran.rstudio.com/")
```

```
sensiti_met_data<-read.table("data/sensitivity_met_data.txt",header=T)%>%na.omit()
str(sensiti_met_data)
```

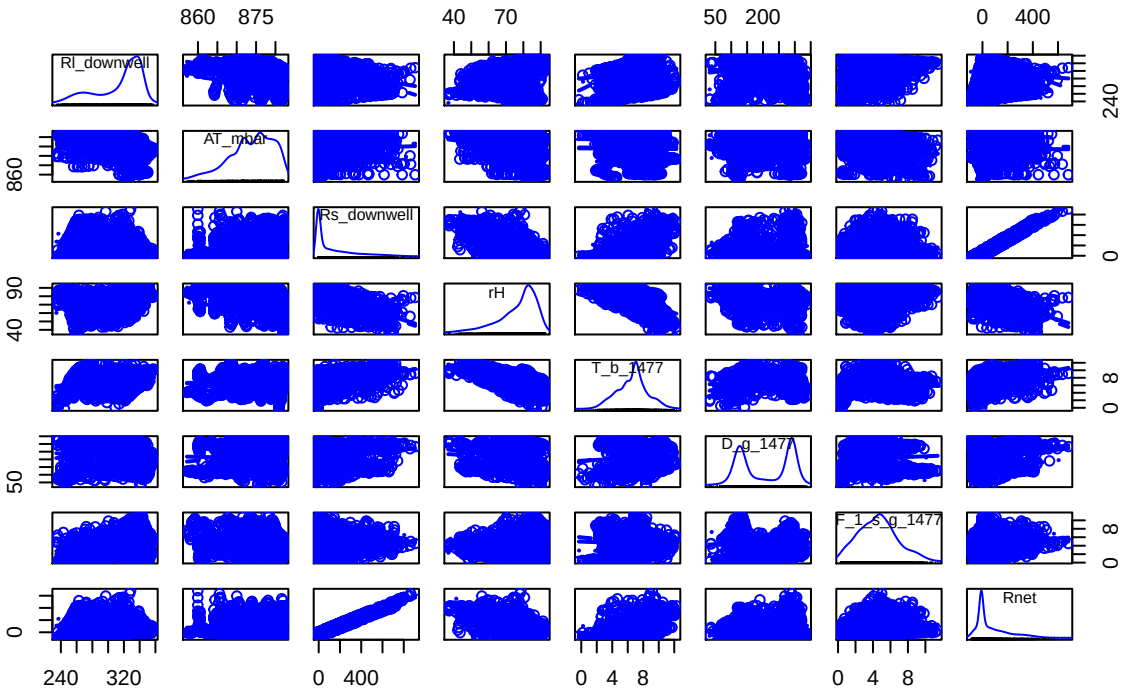
```
## 'data.frame':    1775 obs. of  10 variables:
##  $ X           : int  14755 14756 14757 14758 14759 14760 14761 14762 14763 14764 ...
##  $ date        : Factor w/ 1776 levels "2017-08-03 00:00:00",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Rl_downwell : num  324 326 328 329 326 ...
##  $ AT_mbar     : num  873 873 873 873 872 ...
##  $ Rs_downwell : num  -5.6 -5.28 -5.02 -5.3 -5.78 ...
##  $ rH          : num  81.3 81.3 81.9 82.2 80.6 ...
##  $ T_b_1477    : num  5.57 5.62 5.57 5.69 5.77 ...
##  $ D_g_1477    : num  112 116 132 126 112 ...
##  $ F_1_s_g_1477: num  1.731 0.945 0.955 1.474 1.928 ...
##  $ Rnet        : num  -15.4 -14.9 -13.2 -11.5 -14.1 ...
##  - attr(*, "na.action")= 'omit' Named int 1629
##   ..- attr(*, "names")= chr "1629"
```

```
summary(sensiti_met_data)
```

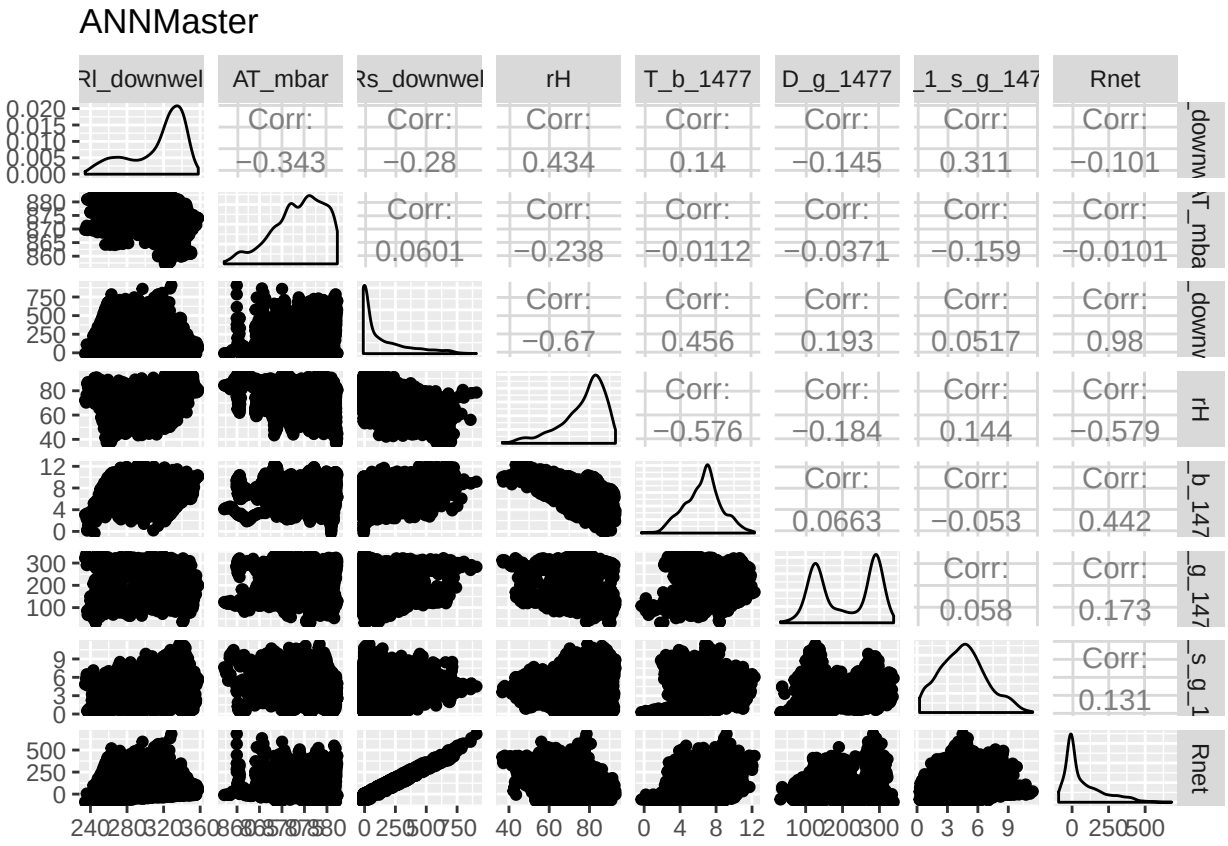
```
##        X                          date       Rl_downwell
##  Min.   :14755   2017-08-03 00:00:00:   1   Min.   :234.2
##  1st Qu.:15198   2017-08-03 00:30:00:   1   1st Qu.:295.9
##  Median :15642   2017-08-03 01:00:00:   1   Median :325.2
##  Mean   :15642   2017-08-03 01:30:00:   1   Mean   :314.1
##  3rd Qu.:16086   2017-08-03 02:00:00:   1   3rd Qu.:337.0
##  Max.   :16530   2017-08-03 02:30:00:   1   Max.   :357.9
##                  (Other)            :1769
##     AT_mbar       Rs_downwell           rH           T_b_1477
##  Min.   :857.2   Min.   :-10.536   Min.   :36.73   Min.   :-0.3046
##  1st Qu.:870.2   1st Qu.: -6.133   1st Qu.:71.44   1st Qu.: 5.2602
##  Median :874.6   Median : 29.409   Median :80.10   Median : 6.7863
##  Mean   :873.8   Mean   :130.600   Mean   :77.10   Mean   : 6.5791
##  3rd Qu.:878.3   3rd Qu.:209.278   3rd Qu.:84.98   3rd Qu.: 7.7514
##  Max.   :882.3   Max.   :906.671   Max.   :92.92   Max.   :12.2805
##
##     D_g_1477       F_1_s_g_1477          Rnet
##  Min.   : 31.79   Min.   : 0.2499   Min.   :-92.02
##  1st Qu.:128.66   1st Qu.: 2.9408   1st Qu.:-12.67
##  Median :209.17   Median : 4.5229   Median : 12.48
##  Mean   :208.70   Mean   : 4.5707   Mean   : 72.71
##  3rd Qu.:289.31   3rd Qu.: 5.9526   3rd Qu.:135.28
##  Max.   :338.51   Max.   :11.4096   Max.   :682.34
##
```

Including a scatterplotMatrix

```
scatterplotMatrix(~Rl_downwell+AT_mbar+Rs_downwell+rH+T_b_1477+D_g_1477+F_1_s_g_1477+Rnet,data=sensiti_
```




```
# with GGally:
ggpairs(sensiti_met_data[c(3,4,5,6,7,8,9,10)],title="ANNMaster")
```




```
images=list.files("../images/",full.names=T)
images
```

```
## character(0)
```

```
names=gsub("[a-zA-Z]|[[:punct:]]"," ",images)
names
```

```
## character(0)
```

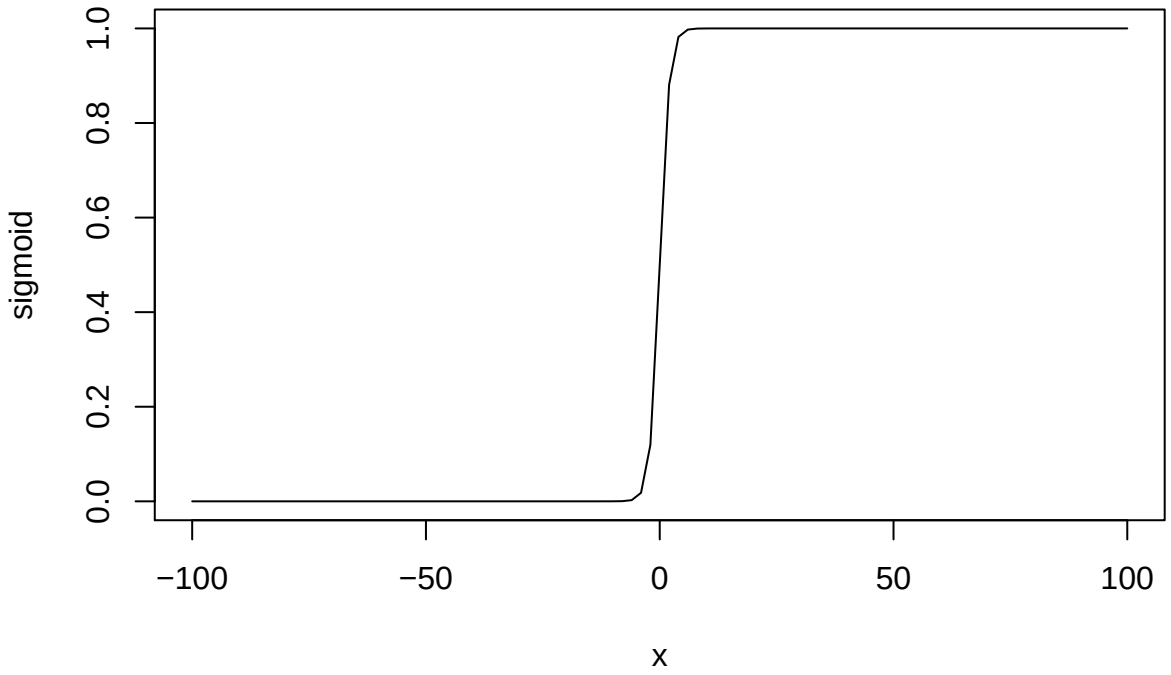
```
#rescale
scale01 <- function(x){
  (x - min(x)) / (max(x) - min(x))
}
sensiti_met_d <- sensiti_met_data[c(3,4,5,6,7,8,9,10)] %>%  mutate_all(scale01)
# Split into test and train sets
set.seed(12345)
#size is the proportion to training
sensiti_Data_Train <- sample_frac(tbl = sensiti_met_d, replace = FALSE, size = 0.80)
sensiti_Data_Test <- anti_join(sensiti_met_d, sensiti_Data_Train)
```

```
## Joining, by = c("Rl_downwell", "AT_mbar", "Rs_downwell", "rH", "T_b_1477", "D_g_1477", "F_1_s_g_1477
```

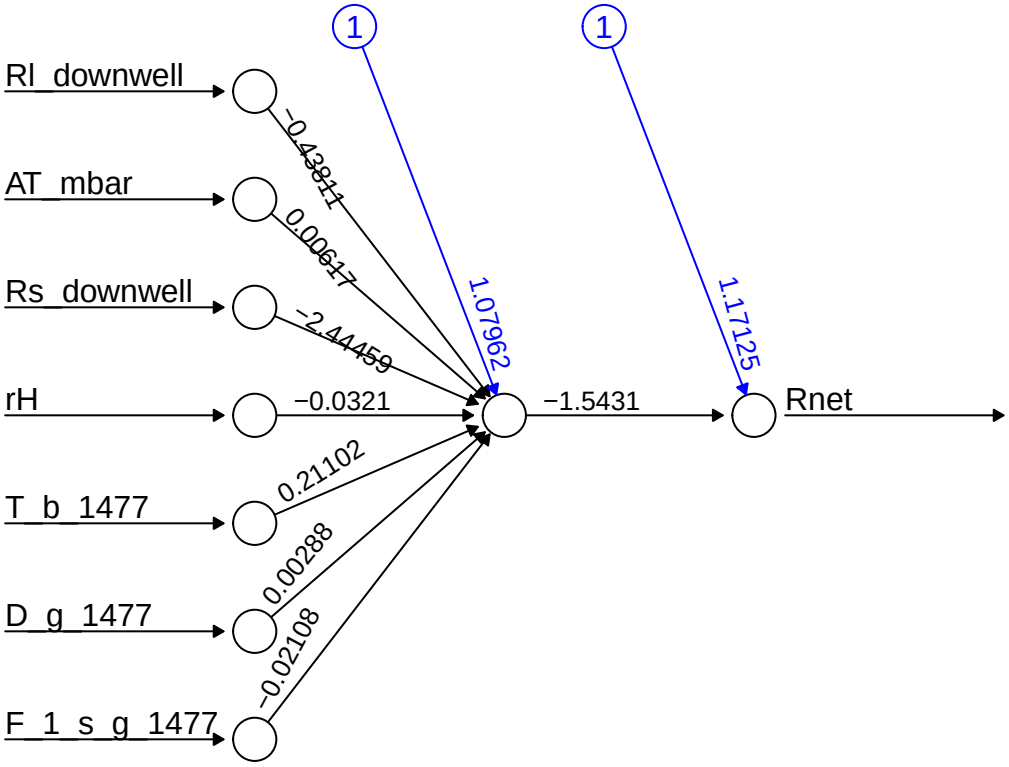
```
?neuralnet
# act.fct = 'logistic' is the sigmoid, 'tanh', default is 'logistic'
#  linear.output = TRUE is to do regression, default
# algorithm = "rprop+" is default (resilient backpropagation with weight backtracking)
```

```
# Calculate activation function
sigmoid <- function(z){1.0/(1.0+exp(-z))}
```

```r
# Partial derivative of activation function
sigmoid_prime <- function(z){sigmoid(z)*(1-sigmoid(z))}
plot(sigmoid, -100.,100.)
```



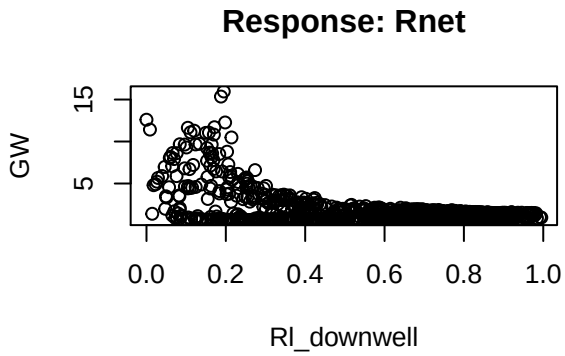

```r
#1-hidden layer ANN with 1 neuron, the simplest of all neural networks:
sensiti_NN1 <- neuralnet(Rnet~Rl_downwell+AT_mbar+Rs_downwell+rH+T_b_1477+D_g_1477+F_1_s_g_1477,
                         data=sensiti_Data_Train)

plot(sensiti_NN1, rep = 'best')
```

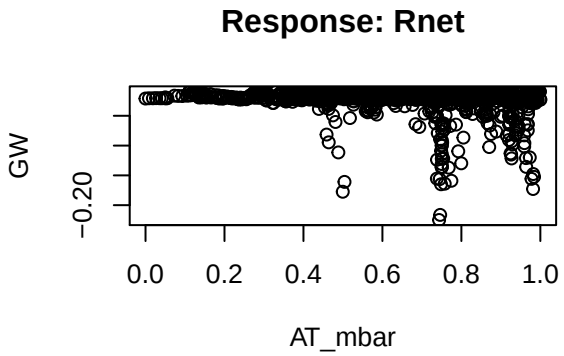


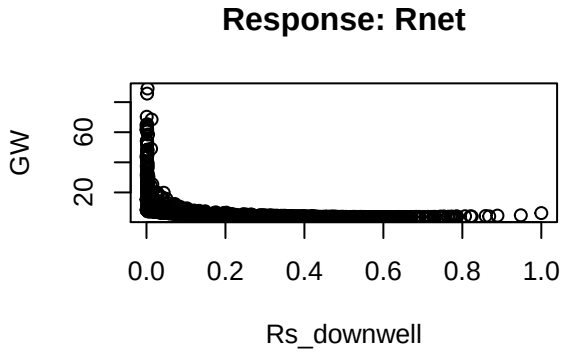
Error: 0.042735 Steps: 24054

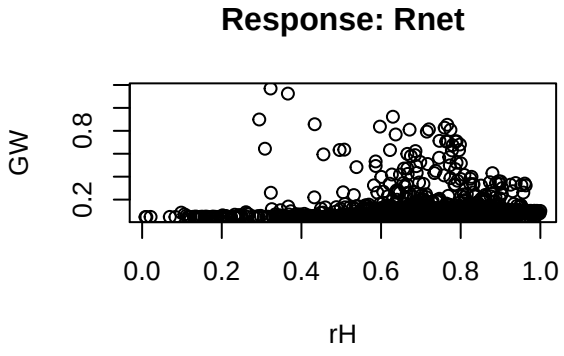
```
# weights learned by the sensiti_NN1 neural network,
# and displays the number of iterations before convergence,
# as well as the SSE (sum of squared errors) of the training data set.
```

```
par(mfrow=c(2,2))  # 2 x 2 pictures on one plot
#gwplot plots the generalized weights for one specific covariate and one response variable
# visualize the generalized weights:
gwplot(sensiti_NN1,selected.covariate="Rl_downwell")
gwplot(sensiti_NN1,selected.covariate="AT_mbar")
gwplot(sensiti_NN1,selected.covariate="Rs_downwell")
gwplot(sensiti_NN1,selected.covariate="rH")
```

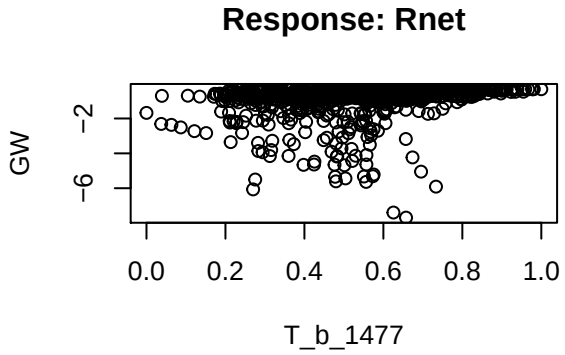


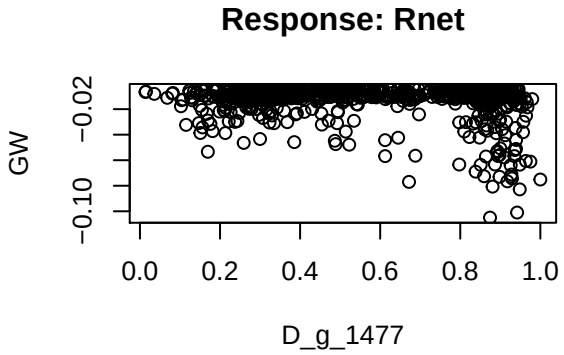


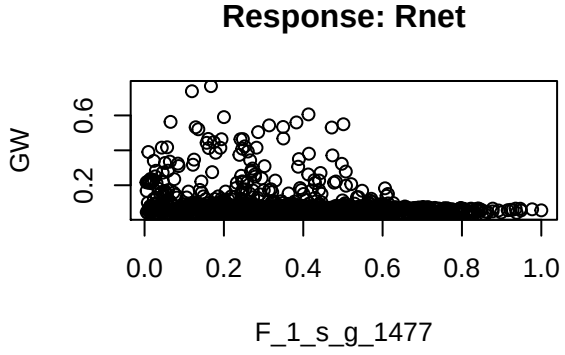





```
gwplot(sensiti_NN1,selected.covariate="T_b_1477")
gwplot(sensiti_NN1,selected.covariate="D_g_1477")
gwplot(sensiti_NN1,selected.covariate="F_1_s_g_1477")
```







```
Test_NN1_Output <- compute(sensiti_NN1, sensiti_Data_Test[, 1:7])$net.result
NN1_Test_SSE <- sum((Test_NN1_Output - sensiti_Data_Test[, 8])^2)/2
NN1_Test_SSE
```

```
## [1] 0.01256679
```

```
# cor to calculate the correlation between the two numeric vectors:
cor(Test_NN1_Output, sensiti_Data_Test[, 8])
```

```
##           [,1]
## [1,] 0.9989776
```

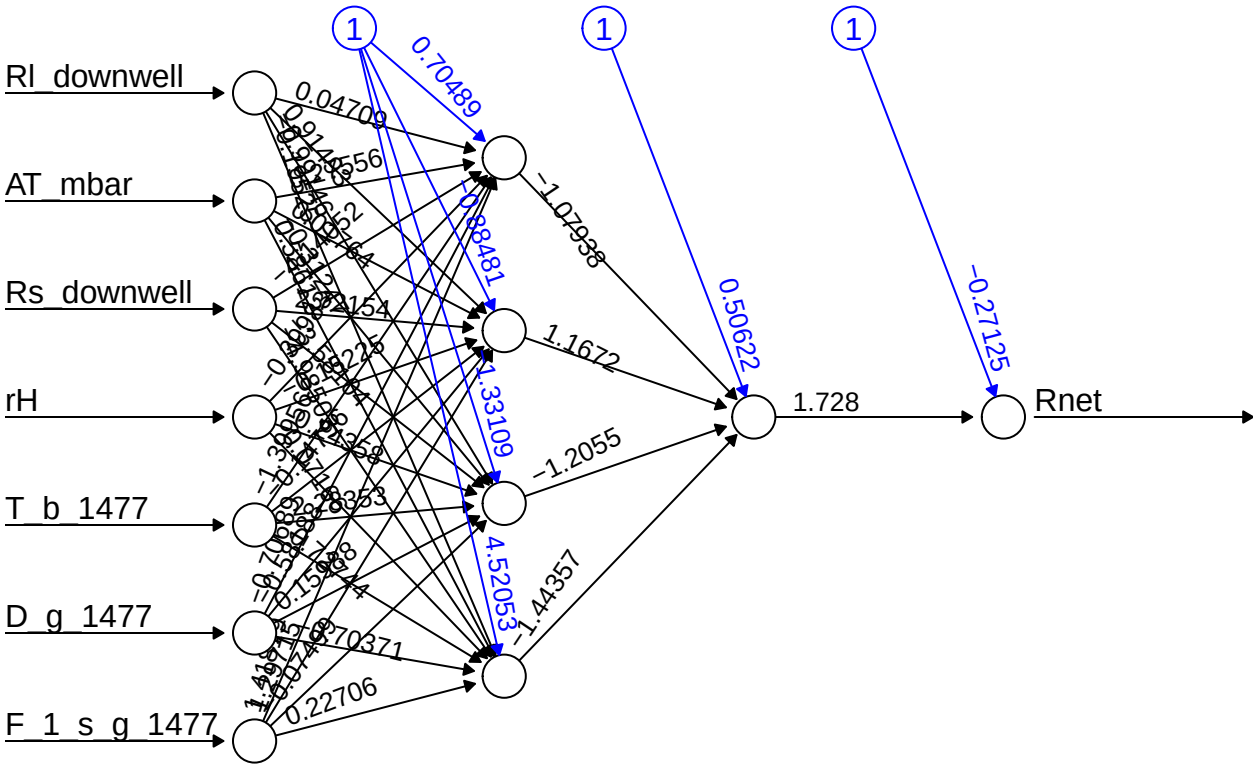
2-Hidden Layers, Layer-1 4-neurons, Layer-2, 1-neuron

```
sensiti_NN2 <- neuralnet(Rnet~Rl_downwell+AT_mbar+Rs_downwell+rH+T_b_1477+D_g_1477+F_1_s_g_1477,
                         data=sensiti_Data_Train,
                         hidden = c(4, 1))
```

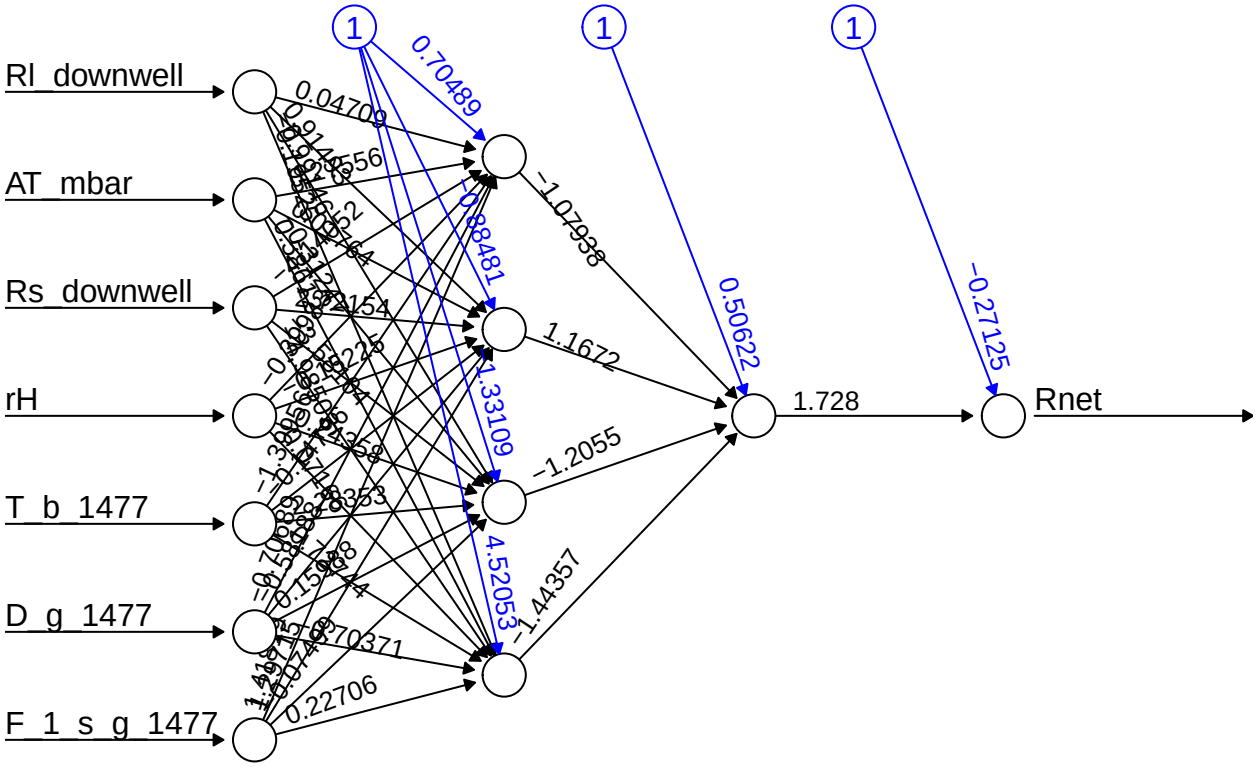
```
## Test Error
Test_NN2_Output <- compute(sensiti_NN2, sensiti_Data_Test[, 1:7])$net.result
NN2_Test_SSE <- sum((Test_NN2_Output - sensiti_Data_Test[, 8])^2)/2
NN2_Test_SSE
```

```
## [1] 0.007706097
```

```
plot(sensiti_NN2, rep = 'best')
```



```
plot(sensiti_NN2, rep = 'best')
```

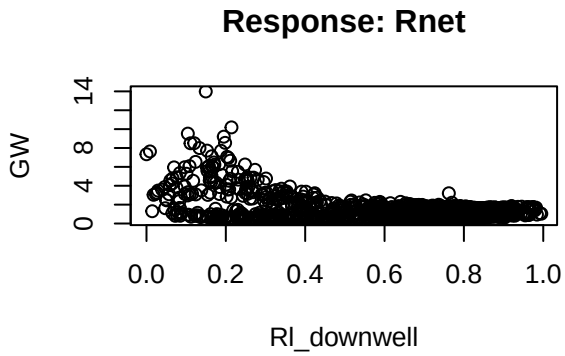


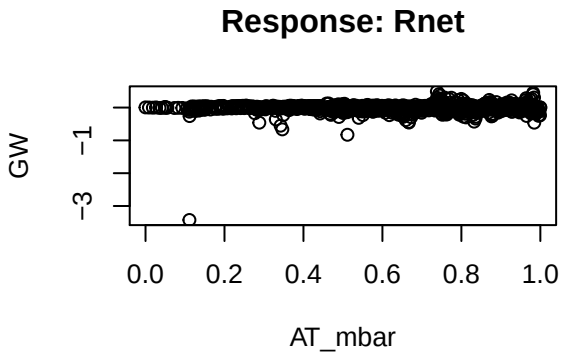
Error: 0.030869 Steps: 1214

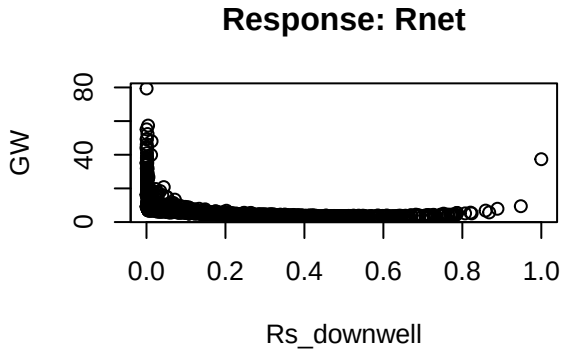
```
# cor to calculate the correlation between the two numeric vectors:
cor(Test_NN2_Output,sensiti_Data_Test[, 8])
```

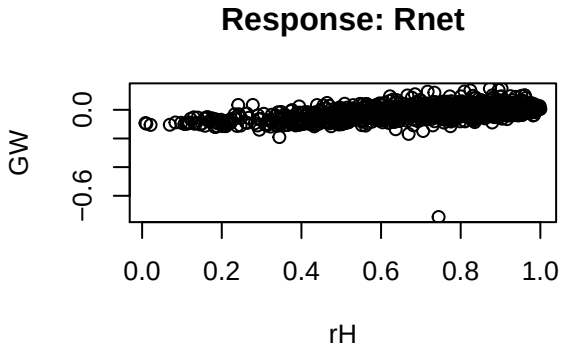
```
##           [,1]
## [1,] 0.9993823
```

```
par(mfrow=c(2,2))
gwplot(sensiti_NN2,selected.covariate="Rl_downwell")
gwplot(sensiti_NN2,selected.covariate="AT_mbar")
gwplot(sensiti_NN2,selected.covariate="Rs_downwell")
gwplot(sensiti_NN2,selected.covariate="rH")
```

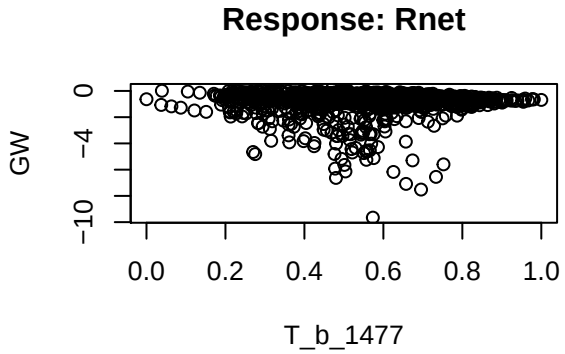


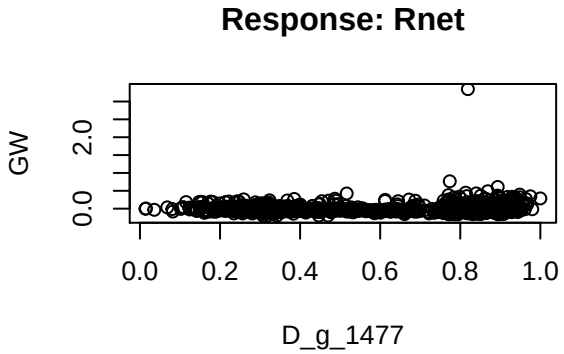


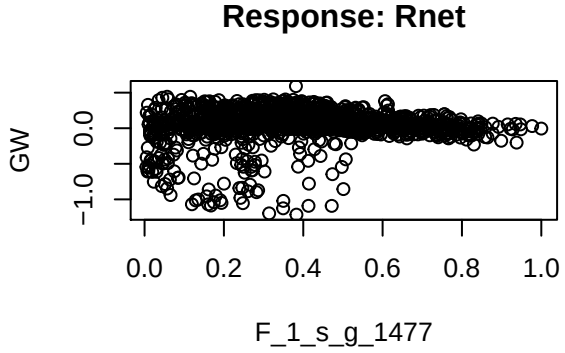





```
gwplot(sensiti_NN2,selected.covariate="T_b_1477")
gwplot(sensiti_NN2,selected.covariate="D_g_1477")
gwplot(sensiti_NN2,selected.covariate="F_1_s_g_1477")
```



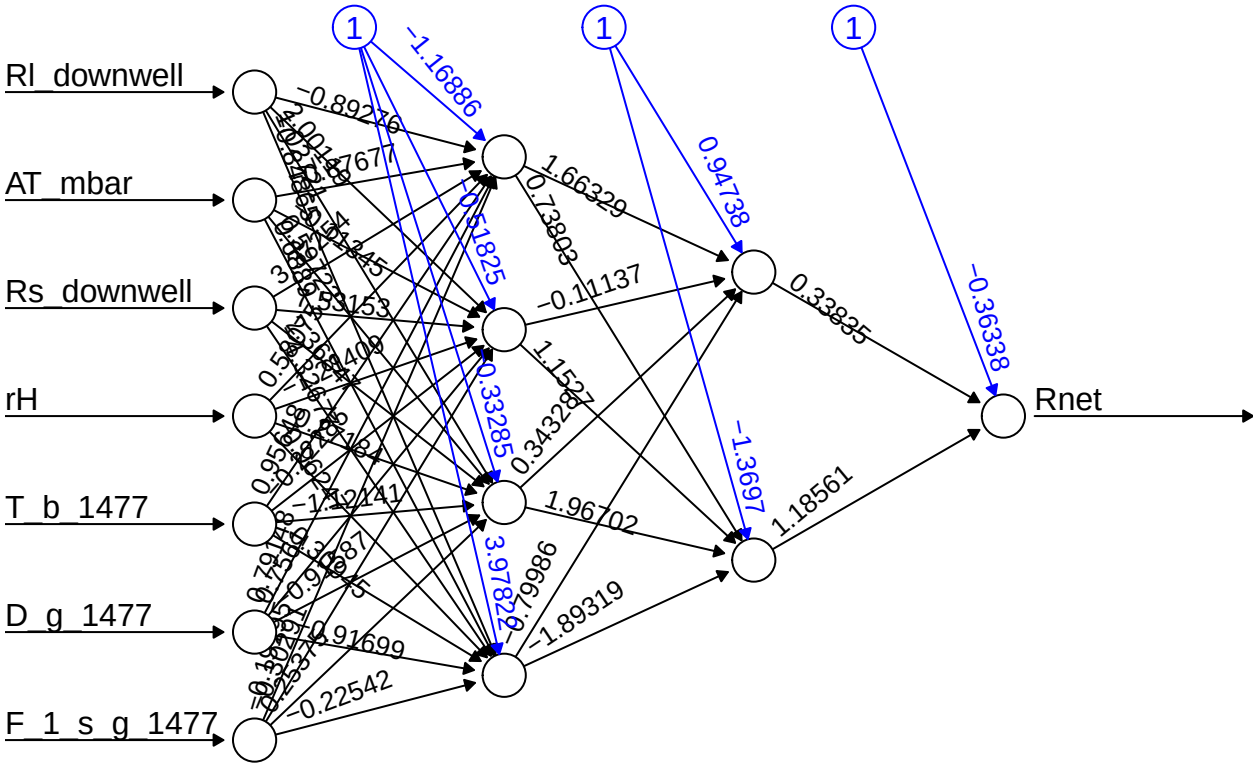




2-Hidden Layers, Layer-1 4-neurons, Layer-2 2-neurons, 1-neuron

```
sensiti_NN3 <- neuralnet(Rnet~Rl_downwell+AT_mbar+Rs_downwell+rH+T_b_1477+D_g_1477+F_1_s_g_1477,data=se
                         hidden = c(4, 2), linear.output = TRUE)
```

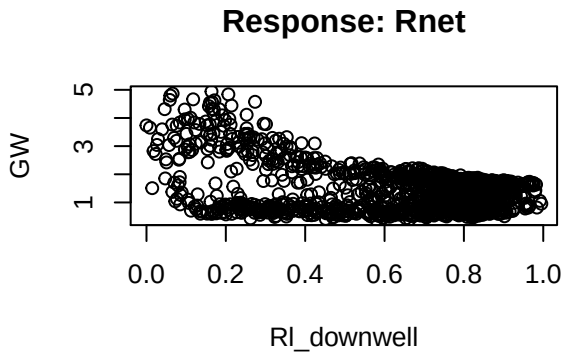
```
## Test Error
Test_NN3_Output <- compute(sensiti_NN3, sensiti_Data_Test[, 1:7])$net.result
NN3_Test_SSE <- sum((Test_NN3_Output - sensiti_Data_Test[, 8])^2)/2

plot(sensiti_NN3, rep = 'best')
```

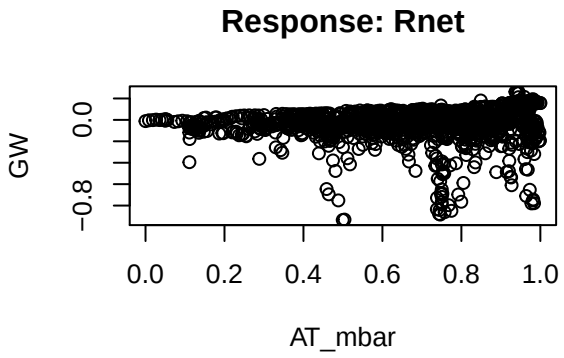


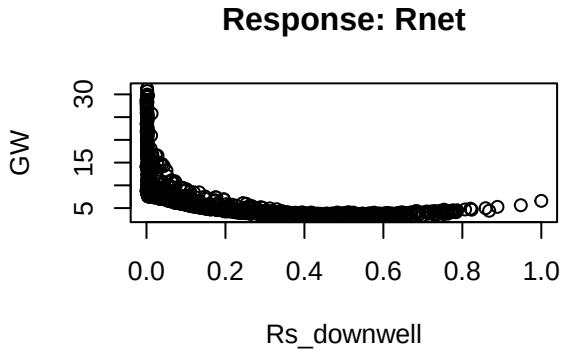

```
# cor to calculate the correlation between the two numeric vectors:
cor(Test_NN3_Output,sensiti_Data_Test[, 8])
```

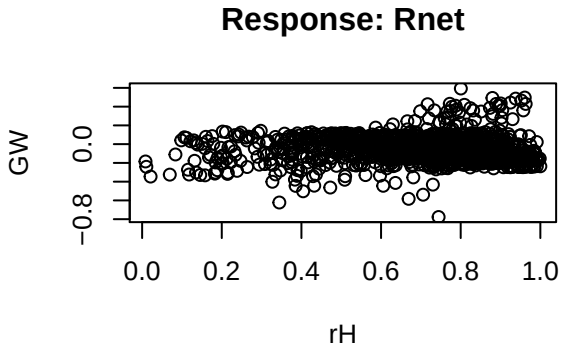
```
##           [,1]
## [1,] 0.9992077
```

```
par(mfrow=c(2,2))
gwplot(sensiti_NN3,selected.covariate="Rl_downwell")
gwplot(sensiti_NN3,selected.covariate="AT_mbar")
gwplot(sensiti_NN3,selected.covariate="Rs_downwell")
gwplot(sensiti_NN3,selected.covariate="rH")
```

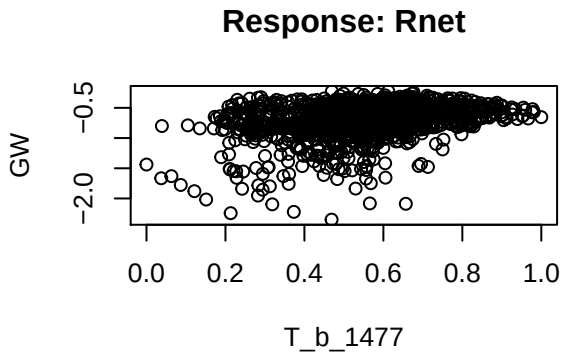


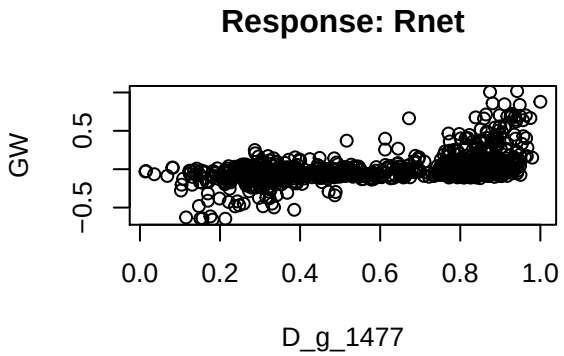


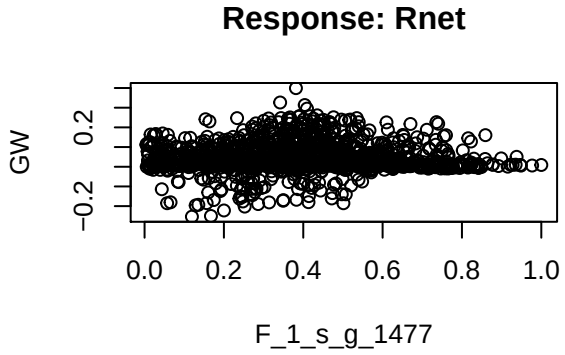




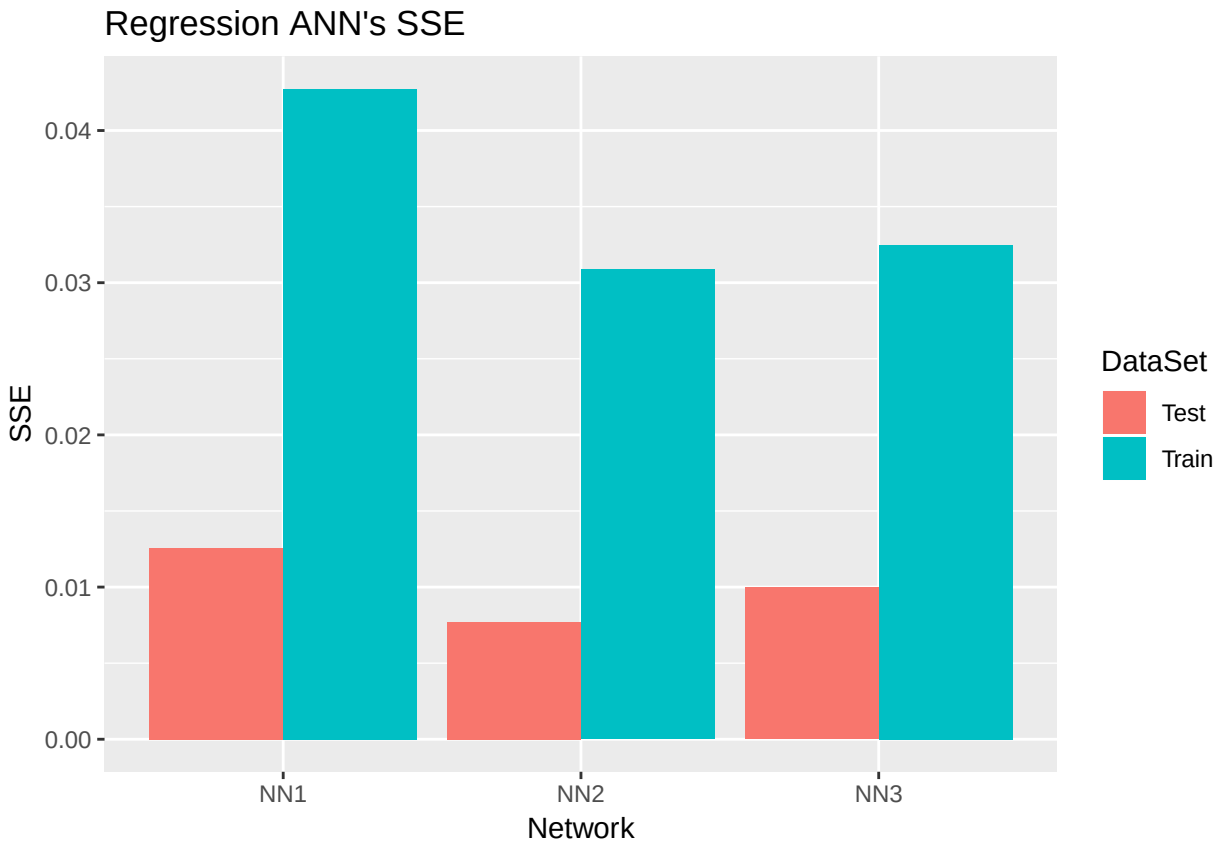

```
gwplot(sensiti_NN3,selected.covariate="T_b_1477")
gwplot(sensiti_NN3,selected.covariate="D_g_1477")
gwplot(sensiti_NN3,selected.covariate="F_1_s_g_1477")
```



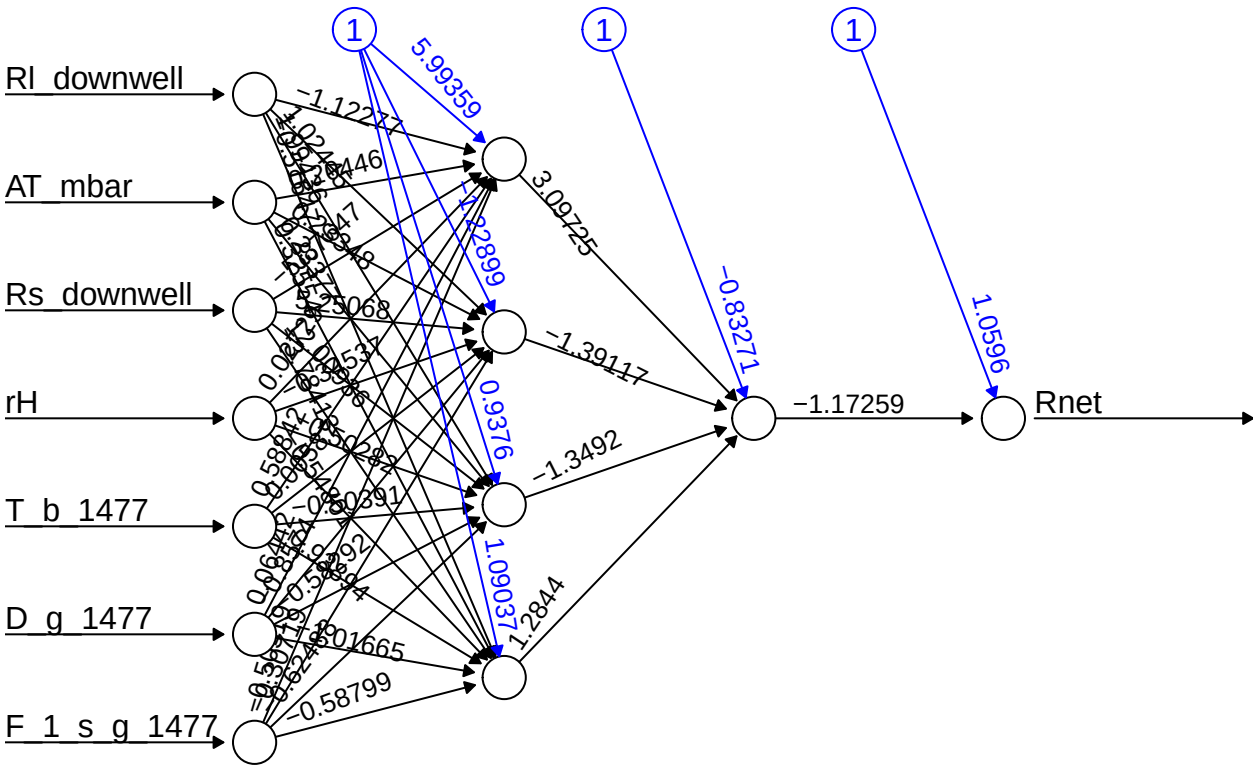




```r
# training error
NN1_Train_SSE <- sensiti_NN1$result.matrix[1]
NN2_Train_SSE <- sensiti_NN2$result.matrix[1]
NN3_Train_SSE <- sensiti_NN3$result.matrix[1]
# Bar plot of results
Regression_NN_Errors <- tibble(Network = rep(c("NN1", "NN2", "NN3"), each = 2),
                               DataSet = rep(c("Train", "Test"), time = 3),
                               SSE = c(NN1_Train_SSE, NN1_Test_SSE,
                                       NN2_Train_SSE, NN2_Test_SSE,
                                       NN3_Train_SSE, NN3_Test_SSE))
Regression_NN_Errors %>% ggplot(aes(Network, SSE, fill = DataSet)) +
  geom_col(position = "dodge") +
  ggtitle("Regression ANN's SSE")
```




```r
# to do regression linear.output=TRUE, it is default
# rep is the repeating the neural network reusing the weights as initial weights
sensiti_NN2 <- neuralnet(Rnet~Rl_downwell+AT_mbar+Rs_downwell+rH+T_b_1477+D_g_1477+F_1_s_g_1477,
                         data=sensiti_Data_Train,
                         hidden = c(4, 1), rep = 10, linear.output=TRUE)
## Test Error
Test_NN2_Output <- compute(sensiti_NN2, sensiti_Data_Test[, 1:7])$net.result
NN2_Test_SSE <- sum((Test_NN2_Output - sensiti_Data_Test[, 8])^2)/2

plot(sensiti_NN2, rep = "best")
```



Error: 0.019969 Steps: 11446

Generalised linear model regression

Now in order to compare with the results of the neural network, we can try a generalised linear model to perform regression.

```
#generalised linear model
sensiti_met_d <- sensiti_met_data[c(3,4,5,6,7,8,9,10)] # %>%  mutate_all(scale01)
```

Split into test and train sets

```
#size is the proportion to training
sensiti_Data_Train_uns <- sample_frac(tbl = sensiti_met_d, replace = FALSE, size = 0.80)
sensiti_Data_Test_uns <- anti_join(sensiti_met_d, sensiti_Data_Train_uns)
```

```
## Joining, by = c("Rl_downwell", "AT_mbar", "Rs_downwell", "rH", "T_b_1477", "D_g_1477", "F_1_s_g_1477
```

```
lm.fit <- glm(Rnet~., data=sensiti_Data_Train_uns)
summary(lm.fit)
```

```
##
## Call:
## glm(formula = Rnet ~ ., data = sensiti_Data_Train_uns)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -34.222   -3.088   -0.172    2.892   28.011
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2.350e+02  2.654e+01  -8.854  < 2e-16 ***
## Rl_downwell   8.946e-01  7.263e-03 123.179  < 2e-16 ***
## AT_mbar      -6.416e-02  2.929e-02  -2.190   0.0287 *
## Rs_downwell   7.463e-01  1.103e-03 676.556  < 2e-16 ***
## rH            1.621e-01  2.477e-02   6.543 8.43e-11 ***
## T_b_1477     -4.497e+00  1.113e-01 -40.397  < 2e-16 ***
## D_g_1477      1.256e-03  1.944e-03   0.646   0.5183
## F_1_s_g_1477  4.794e-01  7.500e-02   6.392 2.23e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 32.7638)
##
##     Null deviance: 25704841  on 1419  degrees of freedom
## Residual deviance:    46262  on 1412  degrees of freedom
## AIC: 8994.6
##
## Number of Fisher Scoring iterations: 2
```

```
pr.lm <- predict(lm.fit,sensiti_Data_Test_uns)
MSE.lm <- sum((pr.lm - sensiti_Data_Test_uns$Rnet)^2)/nrow(sensiti_Data_Test_uns)
# mean squared error (MSE) as a measure of how much our predictions are far away from the real data:
paste("MSE.lm: ", round(MSE.lm, 6))
```

```
## [1] "MSE.lm:  24.811683"
```

```
pr.nn <- compute(sensiti_NN2, sensiti_Data_Test[, 1:7])
## Test Error
#Test_NN2_Output <- compute(sensiti_NN2, sensiti_Data_Test[, 1:7])$net.result
#NN2_Test_SSE <- sum((Test_NN2_Output - sensiti_Data_Test[, 8])^2)/2
```

```
pr.nn_ <- pr.nn$net.result*(max(sensiti_met_data$Rnet)-min(sensiti_met_data$Rnet))+min(sensiti_met_data$
test.r <- (sensiti_Data_Test$Rnet)*(max(sensiti_met_data$Rnet)-min(sensiti_met_data$Rnet))+min(sensiti_m
MSE.nn <- sum((test.r - pr.nn_)^2)/nrow(sensiti_Data_Test)
```

compare both MSEs (from generalised linear model and neural network)

```
print(paste(MSE.lm,MSE.nn))
```

```
## [1] "24.81168309255 24.0807330251539"
```

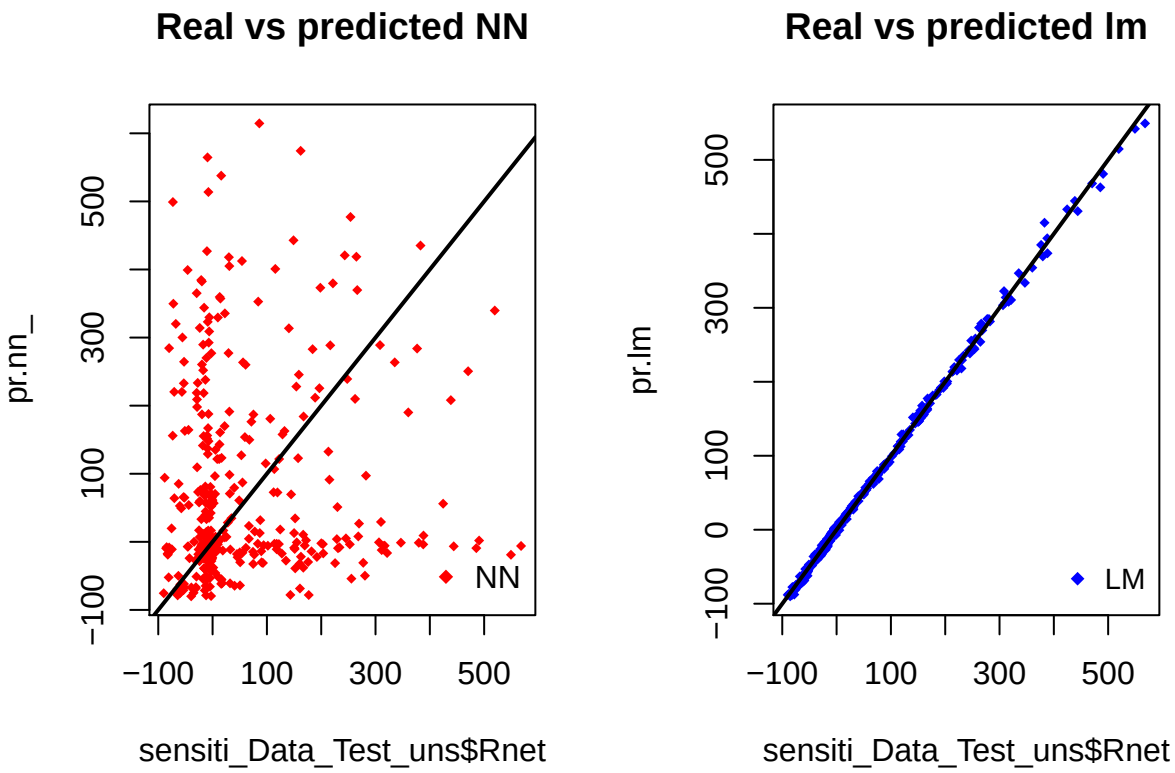
```
MSE.nn < MSE.lm
```

```
## [1] TRUE
```

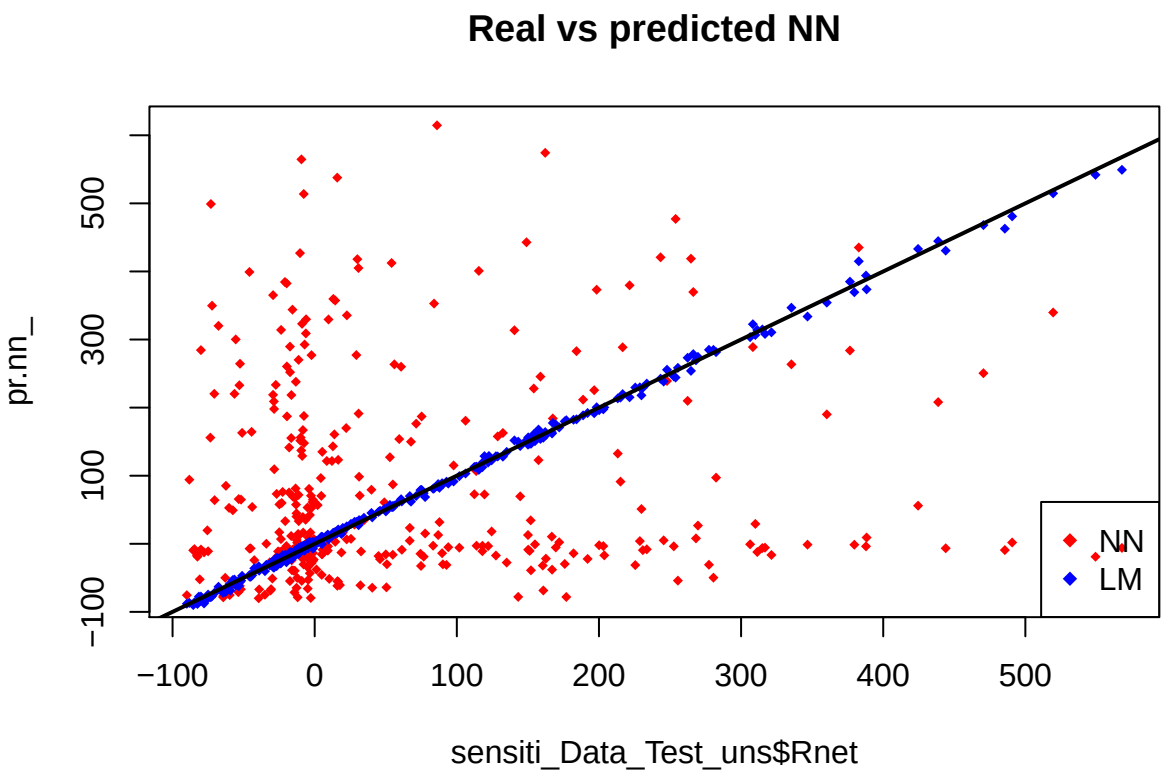
Apparently, the neural network is doing a better work than the linear model at predicting Rnet.

```
par(mfrow=c(1,2))
plot(sensiti_Data_Test_uns$Rnet,pr.nn_,col='red',main='Real vs predicted NN',pch=18,cex=0.7)
```

```
abline(0,1,lwd=2)
legend('bottomright',legend='NN',pch=18,col='red', bty='n')
plot(sensiti_Data_Test_uns$Rnet,pr.lm,col='blue',main='Real vs predicted lm',pch=18, cex=0.7)
abline(0,1,lwd=2)
legend('bottomright',legend='LM',pch=18,col='blue', bty='n', cex=.95)
```




```
plot(sensiti_Data_Test_uns$Rnet,pr.nn_,col='red',main='Real vs predicted NN',pch=18,cex=0.7)
points(sensiti_Data_Test_uns$Rnet,pr.lm,col='blue',pch=18,cex=0.7)
abline(0,1,lwd=2)
legend('bottomright',legend=c('NN','LM'),pch=18,col=c('red','blue'))
```


Real vs predicted NN
pr.nn_
−100
100
300
500
−100
0
100
200
300
400
500
sensiti_Data_Test_uns$Rnet
NN
LM