Data Mining Practice - Neural Network

1. nnet

1.1. Install packages - nnet

```
install.packages("nnet")
```

```
## Installing package into 'C:/Users/dox/Documents/R/win-library/3.0'
## (as 'lib' is unspecified)
```

```
## Error: trying to use CRAN without setting a mirror
```

```
library(nnet)
```

1.2. Data Preparation - IRIS

```
data(iris)
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1
 1 1 1 1 1 ...
```

```
ir <- iris[, -5]

# Train & Test Data Set
set.seed(1234)
ir.ind <- c(sample(1:50, 25), sample(51:100, 25), sample(101:150, 25))
ir.train <- ir[ir.ind, ]
ir.test <- ir[-ir.ind, ]

targets <- class.ind(iris[, 5])
```

1.3. Neural Network

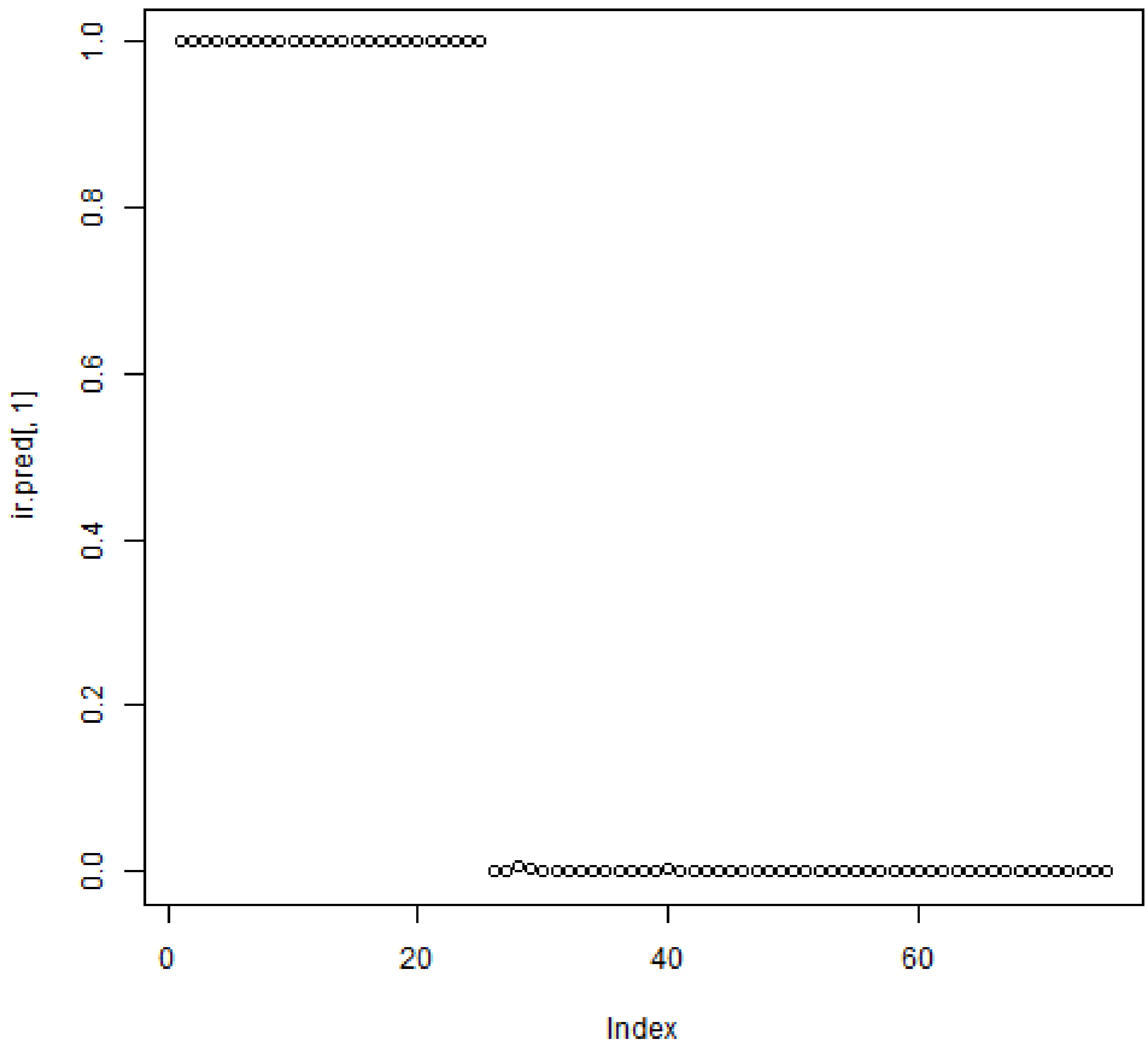
```
# nnet
ir.nn <- nnet(ir.train, targets[ir.ind, ], size = 2, rang = 0.1, softmax =
 T,
    decay = 5e-04, maxit = 200)
```

```
## # weights:  19
## initial  value 82.464734
## iter  10 value 35.842708
## iter  20 value 3.506733
## iter  30 value 1.104380
## iter  40 value 0.750380
## iter  50 value 0.624881
## iter  60 value 0.469364
## iter  70 value 0.417493
## iter  80 value 0.380721
## iter  90 value 0.371818
## iter 100 value 0.363942
## iter 110 value 0.362560
## iter 120 value 0.361618
## iter 130 value 0.361396
## iter 140 value 0.361311
## iter 150 value 0.361303
## final  value 0.361298
## converged
```

```
summary(ir.nn)
```

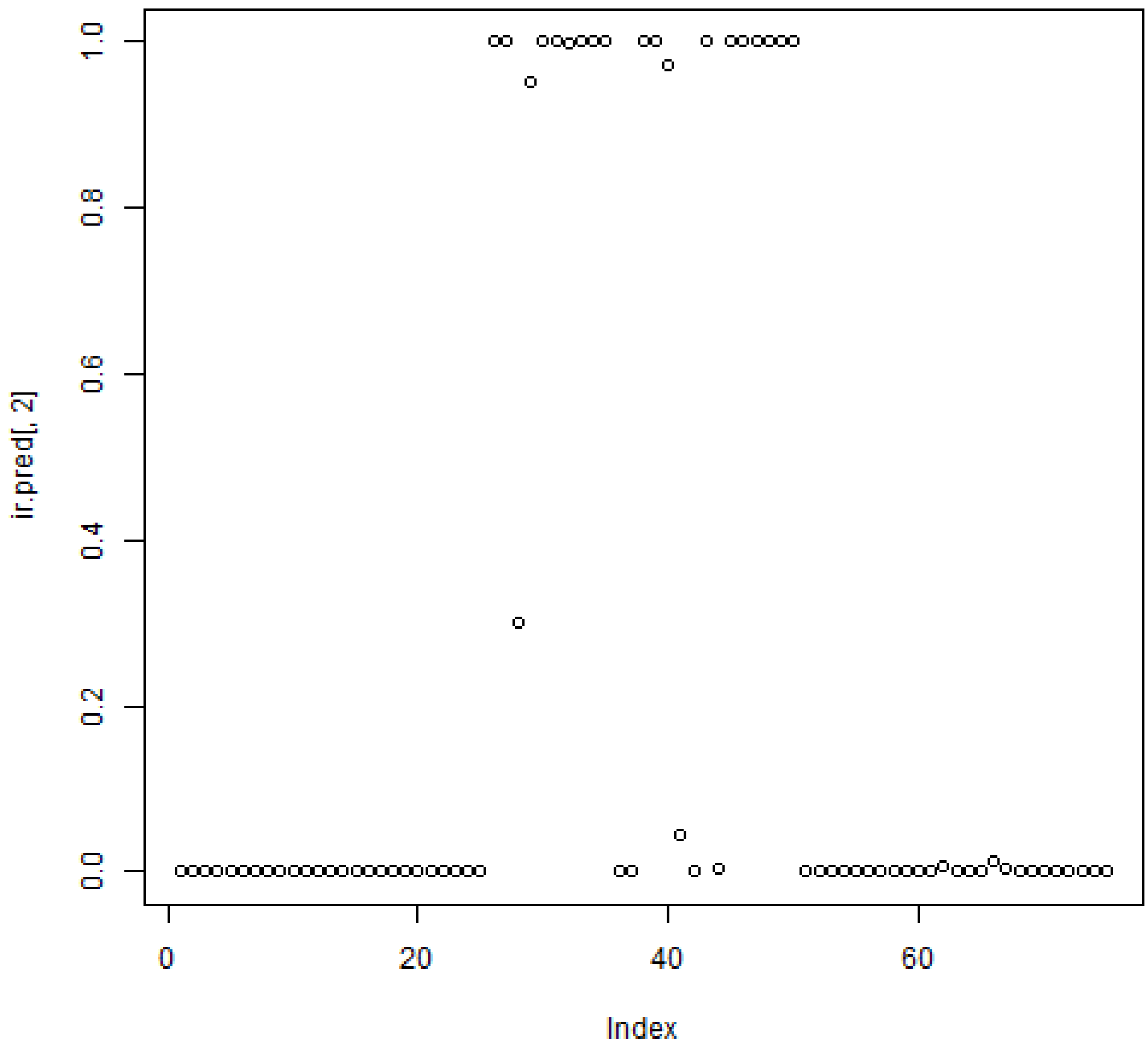
```
## a 4-2-3 network with 19 weights
## options were - softmax modelling  decay=5e-04
##  b->h1 i1->h1 i2->h1 i3->h1 i4->h1
##   3.63   0.23   9.04  -3.85  -8.14
##  b->h2 i1->h2 i2->h2 i3->h2 i4->h2
##   0.37   0.62   1.75  -2.95  -1.26
##  b->o1 h1->o1 h2->o1
##  -3.37   0.87   8.87
##  b->o2 h1->o2 h2->o2
##  -2.87   9.32  -8.26
##  b->o3 h1->o3 h2->o3
##   6.24 -10.19  -0.61
```

```
# predict
ir.pred <- predict(ir.nn, ir.test)

# Prediction vs. Actual
con.nn.table <- table(actual = iris[-ir.ind, 5], pred = predict(ir.nn, ir.
test,
    type = "class"))

plot(ir.pred[, 1])
```



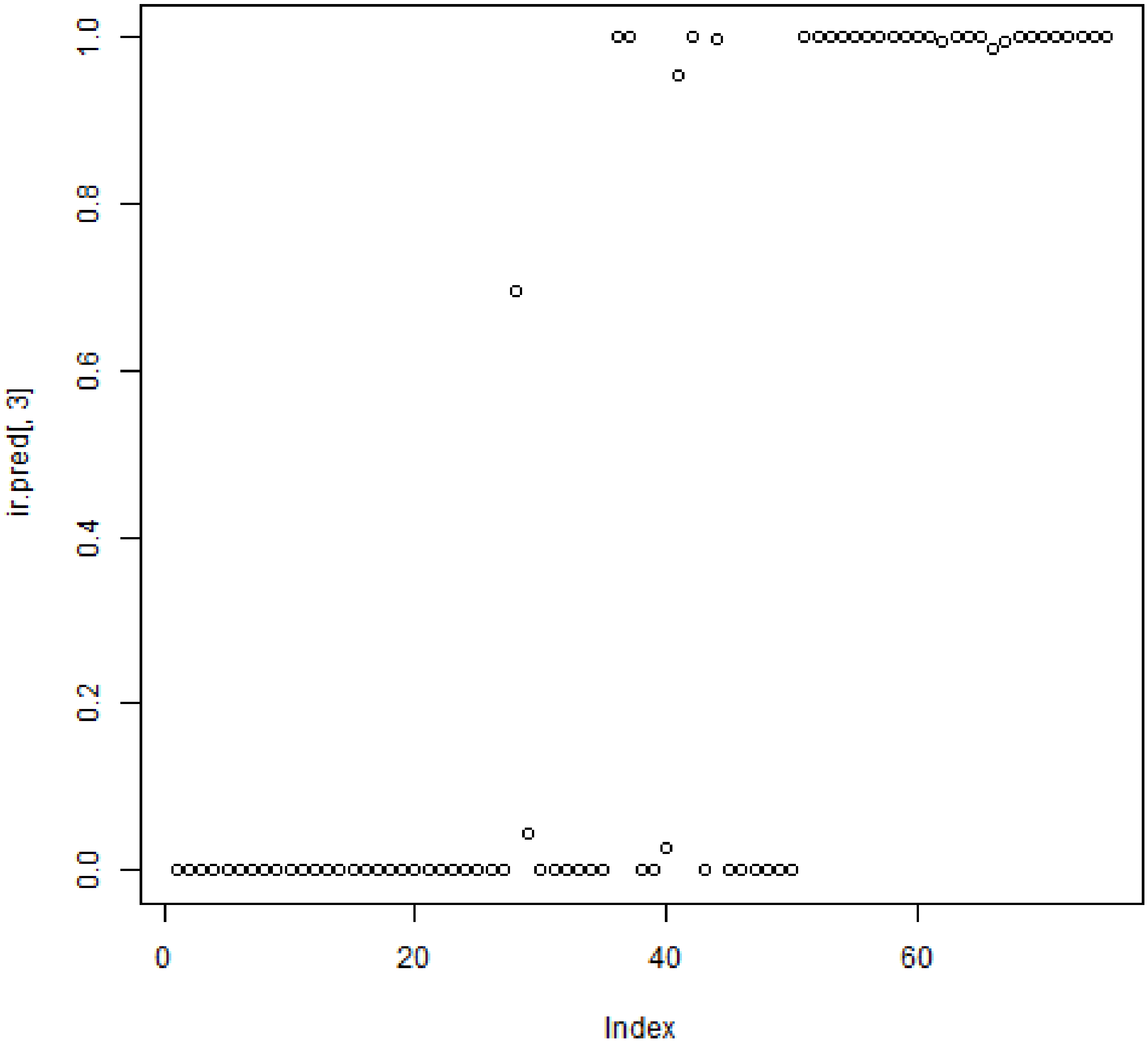
plot of chunk unnamed-chunk-3

```
plot(ir.pred[, 2])
```



plot of chunk unnamed-chunk-3

```
plot(ir.pred[, 3])
```



plot of chunk unnamed-chunk-3

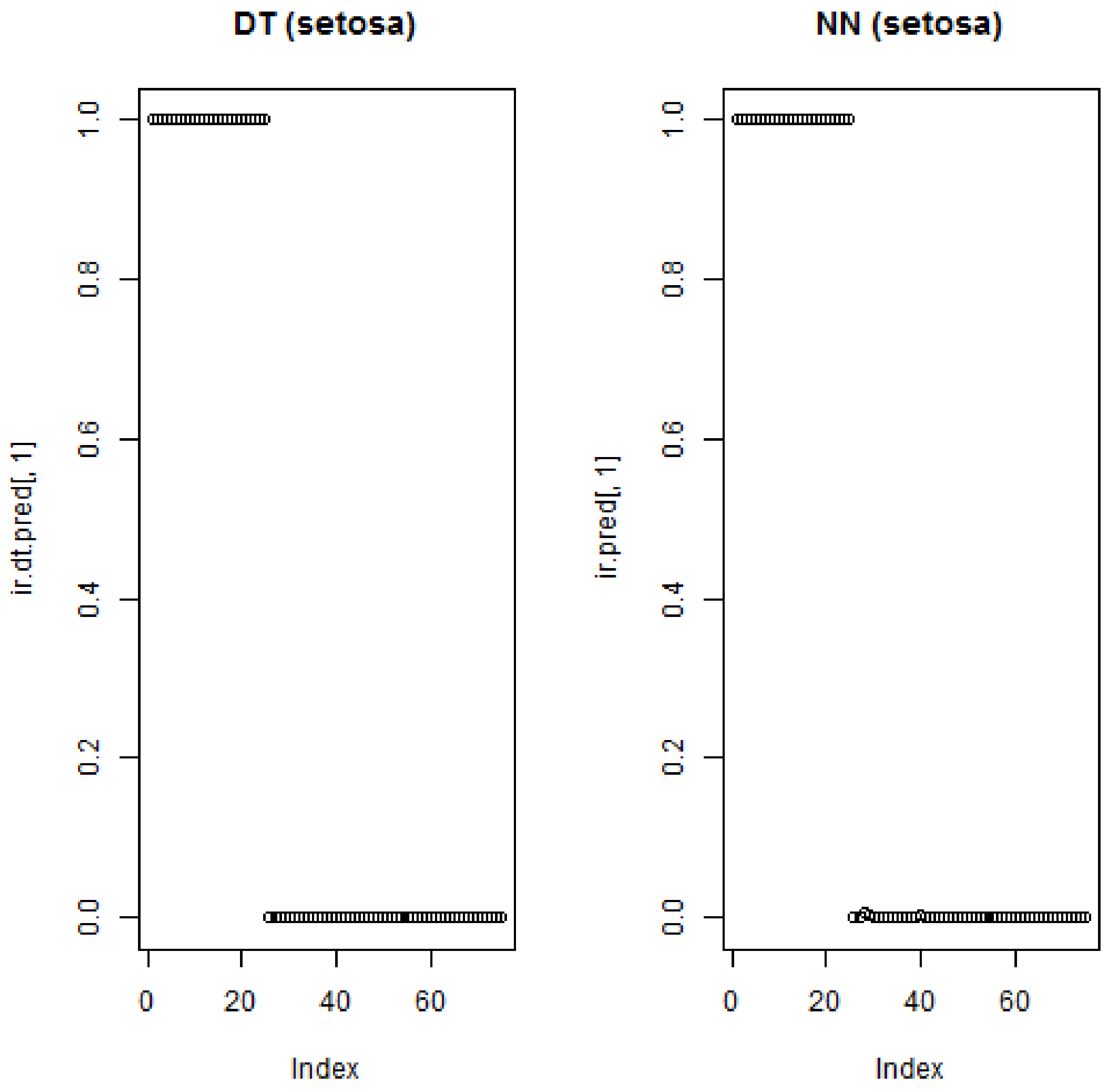
1.4. Compare with Tree Model

```
library(rpart)

ir.dt <- rpart(iris[-ir.ind, 5] ~ ., data = ir.train, method = "class", co
ntrol = rpart.control(xval = 50))
# predict
ir.dt.pred <- predict(ir.dt, ir.test)

# Prediction vs. Actual
con.dt.table <- table(actual = iris[-ir.ind, 5], pred = predict(ir.dt, ir.
```

```r
test,
    type = "class"))

list(nn = con.nn.table, dt = con.dt.table)
```

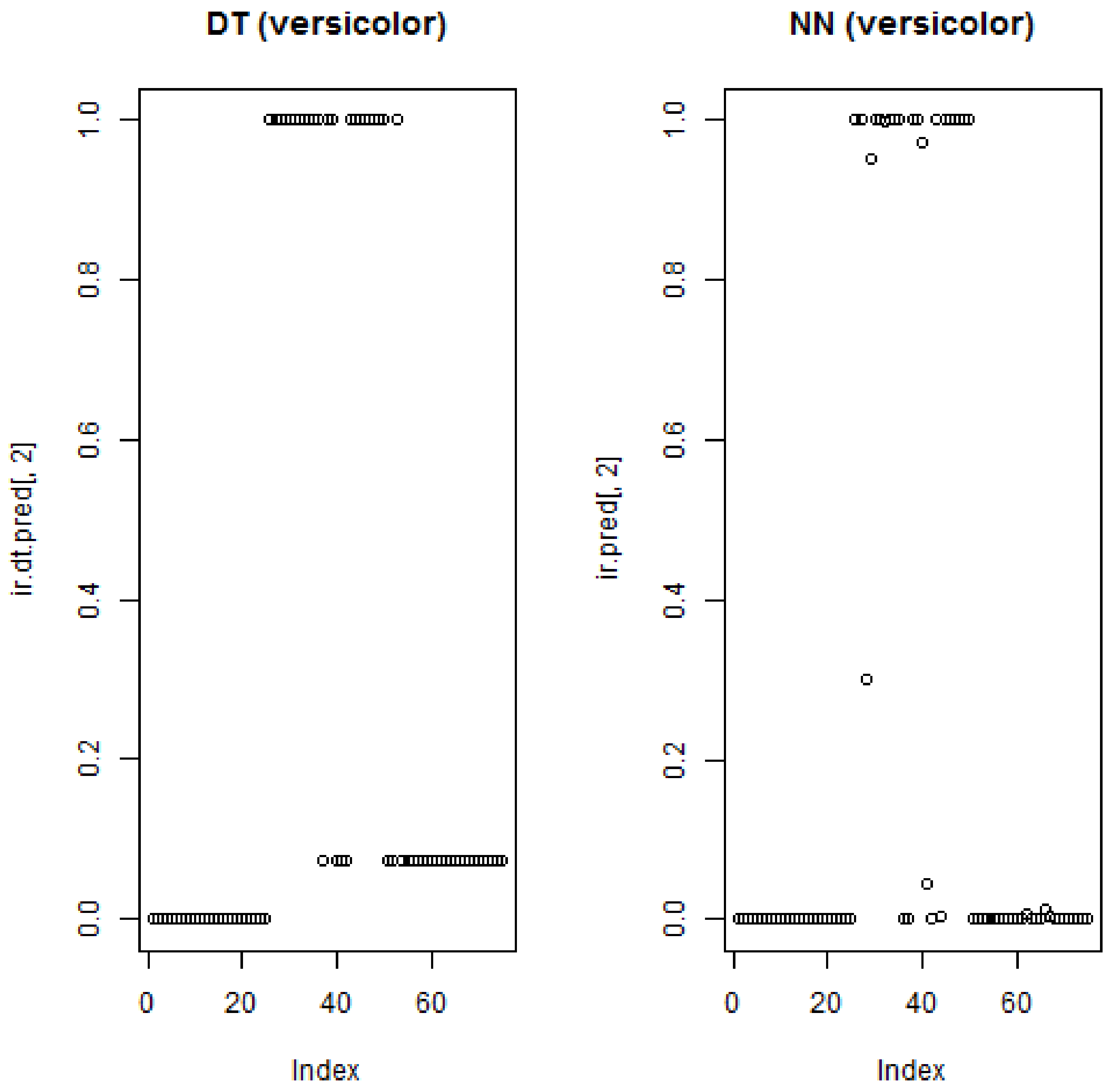
```
## $nn
##             pred
## actual       setosa versicolor virginica
##   setosa         25          0         0
##   versicolor      0         19         6
##   virginica       0          0        25
##
## $dt
##             pred
## actual       setosa versicolor virginica
##   setosa         25          0         0
##   versicolor      0         21         4
##   virginica       0          1        24
```

```r
par(mfrow = c(1, 2))
plot(ir.dt.pred[, 1], main = "DT (setosa)")
plot(ir.pred[, 1], main = "NN (setosa)")
```



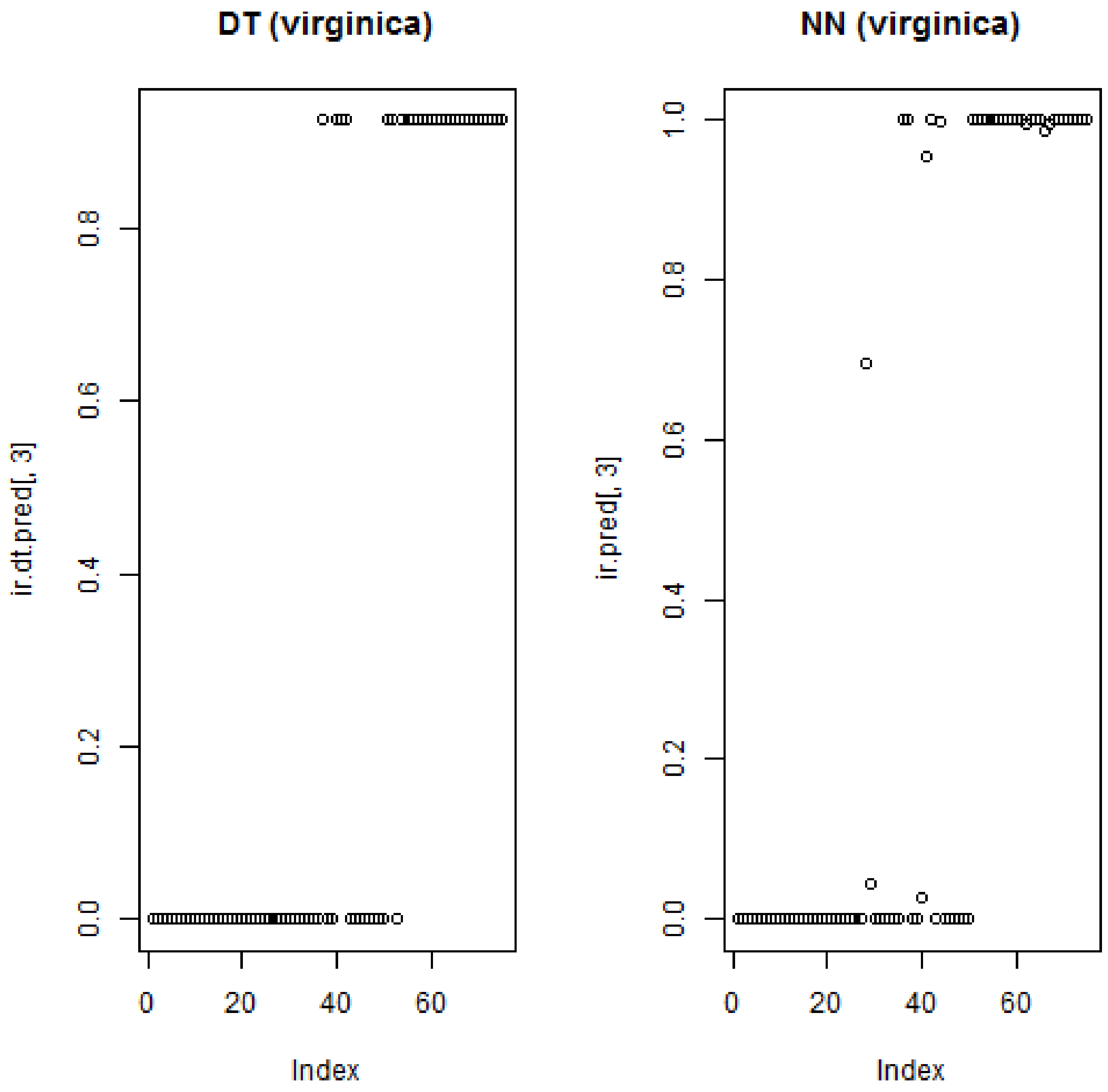
plot of chunk unnamed-chunk-4

```
plot(ir.dt.pred[, 2], main = "DT (versicolor)")
plot(ir.pred[, 2], main = "NN (versicolor)")
```



plot of chunk unnamed-chunk-4

```
plot(ir.dt.pred[, 3], main = "DT (virginica)")
plot(ir.pred[, 3], main = "NN (virginica)")
```



plot of chunk unnamed-chunk-4

2. neuralnet

```
library(neuralnet)
```

```
## Loading required package: grid
## Loading required package: MASS
```

```
names(infert)
```

```
## [1] "education"      "age"            "parity"         "induced"
## [5] "case"           "spontaneous"    "stratum"        "pooled.stratum"
```

```
data(infert)

nn <- neuralnet(case ~ age + parity + induced + spontaneous, data = infert,

    hidden = 2, err.fct = "ce", linear.output = FALSE)
```

```
names(nn)
```

```
##  [1] "call"                "response"            "covariate"
##  [4] "model.list"          "err.fct"             "act.fct"
##  [7] "linear.output"       "data"                "net.result"
## [10] "weights"             "startweights"        "generalized.weights"
## [13] "result.matrix"
```

```
nn$result.matrix
```

```
##                                          1
## error                     119.601948106473
## reached.threshold           0.008795874394
## steps                   84188.000000000000
## Intercept.to.1layhid1      -6.495031550523
## age.to.1layhid1             0.061576891810
## parity.to.1layhid1         -1.206864948136
## induced.to.1layhid1         1.510769090307
## spontaneous.to.1layhid1     2.519563518710
## Intercept.to.1layhid2      -3.664457216303
## age.to.1layhid2            -4.575183919458
## parity.to.1layhid2         10.514939299055
## induced.to.1layhid2      -759.978565706809
## spontaneous.to.1layhid2    54.852790743434
## Intercept.to.case          -1.842169741842
## 1layhid.1.to.case          63.726989544858
## 1layhid.2.to.case          -4.777580217870
```

```
out <- cbind(nn$covariate, nn$net.result[[1]])

dimnames(out) <- list(NULL, c(names(infert)[c(2, 3, 4, 6)], "nn-output"))

head(nn$generalized.weights[[1]])
```

```
##               [,1]           [,2]          [,3]          [,4]
## 1 0.0146154312646 -0.28645245281 0.35858487083 0.59802478401
## 2 0.1011235079641 -1.98195156669 2.48103250497 4.13770643599
## 3 0.0009619624101 -0.01885380506 0.02360143607 0.03936095707
## 4 0.0078737218519 -0.15431956104 0.19317921464 0.32217186921
## 5 0.0741351490444 -1.45299819749 1.81888186282 3.03341418341
## 6 0.1050508399238 -2.05892458599 2.57738832219 4.29840247051
```

```r
# visualization
plot(nn)
```