

```

#### Custom Neuralnet - Iris Dataset

### Setup
set.seed(12345)

## Iris dataset
data.iris = as.data.frame(iris)
idx = data.iris$Species %in% c("virginica", "versicolor")
y.iris = ifelse(data.iris[idx,]$Species=="virginica", 1, 0)
X.iris = data.iris[idx,c(1,3)] # Select only Lengths to be able to
plot
n.iris = dim(X.iris)[1]
X.iris = scale(X.iris)

# Data info
n.all = n.iris
X.all = X.iris
y.all = y.iris

# Split into training and validation set
ids = sample(1:n.all, n.all/2)
train = X.all[ids,] # training
valid = X.all[-ids,] # validation
X.train = as.matrix(X.all[ids,])
X.valid = as.matrix(X.all[-ids,])
y.train = y.all[ids]
y.valid = y.all[-ids]
n.train = dim(X.train)[1]
n.valid = dim(X.valid)[1]

# init weights
w.k = as.matrix(runif(2, -0.5, 0.5), ncol=1, nrow=2)
b.k = runif(1, -0.5, 0.5)

learning.rate = 0.5
n.iterations = 1000 #number of training iterations
error.train = numeric()
error.valid = numeric()

### Functions
sigmoid = function(z){
  return(1 / (1+exp(-z)))
}

### Implementation

for(n in 1:n.iterations){
  # error train
  a.k.train = X.train %*% w.k + b.k
  y.k.train = sigmoid(a.k.train) > 0.5
  error.train[n] = mean(abs(y.k.train-y.train))

  # error validation
  a.k.valid = X.valid %*% w.k + b.k

```

```

y.k.valid = sigmoid(a.k.valid) > 0.5
error.valid[n] = mean(abs(y.k.valid-y.valid))

cat("n: ", n, "| error.train: ", error.train[n], "| error.valid: ",
    error.valid[n], "\n")
flush.console()

for(i in 1:n.train){
  ## forward propagation
  a.k = X.train[i,] %*% w.k + b.k
  y.k = sigmoid(a.k)[1]

  ## backward propagation
  # calculate delta
  d.k = (y.k-y.train[i]) * y.k * (1-y.k)

  # update weights and biases
  b.k = b.k - learning.rate*d.k
  w.k = w.k - learning.rate*d.k*X.train[i,]
}
}

# Plot errors
plot(error.valid[-1], type="l", col=3, ylim=c(0, 0.5))
lines(error.train[-1], type="l", col=1)

# Fit data
yfit.train = ifelse(y.k.train>0.5, 1, 0)
yfit.valid = ifelse(y.k.valid>0.5, 1, 0)

# Training set
plot(X.train[,1], X.train[,2], col=rgb(y.train,1-y.train,0,0.5),
     pch=19,
     xlab="X", ylab="Y")
points(X.train[,1], X.train[,2], col=rgb(yfit.train,1-
yfit.train,0,0.5), pch=3)

# Validation set
plot(X.valid[,1], X.valid[,2], col=rgb(y.valid,1-y.valid,0,0.5),
     pch=19,
     xlab="X", ylab="Y")
points(X.valid[,1], X.valid[,2], col=rgb(yfit.valid,1-
yfit.valid,0,0.5), pch=3)

```