Assignment 4 - Recognition of Handwritten Digits

Hugh Jamieson, jamieson.65

```
setwd("/Users/hughj/Development/osu/osu-mach-learn/module-4")
image_data <- as.matrix(read.csv("image_data.txt", sep = ';', header = FALSE))
labels <- as.matrix(read.csv("labels.txt", header = FALSE))
w1<-as.matrix(read.csv("W1.txt", header = FALSE, sep = ";"))
w2 <- as.matrix(read.csv("W2.txt", header = FALSE, sep = ";"))

dim(image_data)
```

```
## [1] 2500  400
```

```
dim(labels)
```

```
## [1] 2500    1
```

```
dim(w1)
```

```
## [1]  20 401
```

```
dim(w2)
```

```
## [1]  5 21
```

```
#(c) implement feedforward propagation
sigma <- function(z){
  1 / (1 + exp(-z))
}
neural.net <-function(im){
  s1<-matrix(im,nrow=1,ncol=400)
  s1b<-rbind(1, t(s1))
  s2 <- sigma(w1 %*% s1b)
  s2b<-rbind(1, s2)
  s3 <- sigma(w2 %*% s2b)
  s3
}
```

predict a sample row

```
sample.row.test<-neural.net(image_data[1,])
t(sample.row.test)
```

```
##          [,1]      [,2]        [,3]        [,4]        [,5]
## [1,] 0.458538 0.1584544 0.003605214 0.001742082 0.001666637
```

```
# select the highest value
which(sample.row.test==max(sample.row.test))
```

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

Process all 2500 images and summarize our prediction output

```
output.layer<- t(apply(image_data,1,neural.net))
prediction<-apply(output.layer, 1, function(nodes){which(nodes == max(nodes))})
# print count of each estimated images
library(plyr)
count(prediction)
```

```
##   x freq
## 1 1  409
## 2 2  565
## 3 3  530
## 4 4  490
## 5 5  506
```

Summarize the labels data (for reference)

```
count(labels)
```

```
##   V1 freq
## 1  1  500
## 2  2  500
## 3  3  500
## 4  4  500
## 5  5  500
```

(d) Calculate percent accuracy (correct predictions/total images)

```
table(labels, prediction)
```

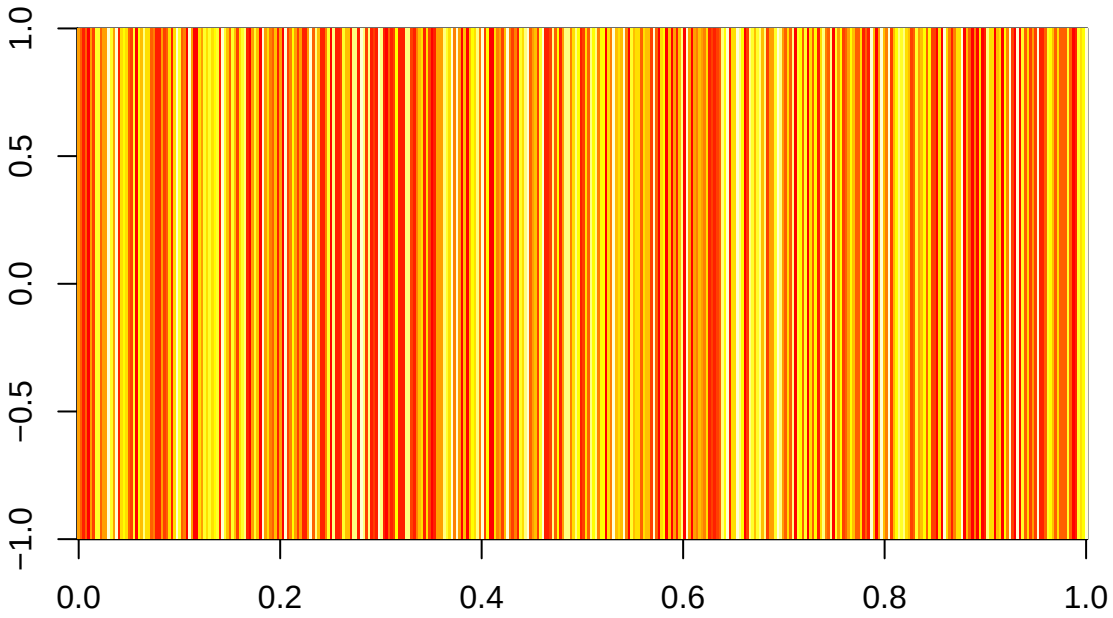
```
##       prediction
## labels   1   2   3   4   5
##      1 408  56  32   0   4
##      2   0 490   2   3   5
##      3   0   7 481   1  11
##      4   1   9   3 485   2
##      5   0   3  12   1 484
```

```
pct.accuracy<-colMeans(prediction==labels)
cat(sprintf("Prediction accuracy=%f", pct.accuracy*100))
```

```
## Prediction accuracy=93.920000
```

(e) Generate random image and classlify

```
# (e) generate a random image and plot
test.image = as.matrix(runif(400))
image(test.image)
```




```
# test.image - feed to neural net
img.e<-neural.net(test.image)
cat(sprintf("number prediction = %d",which(img.e == max(img.e))))
```

```
## number prediction = 3
```

(i)

The network defines nonlinear boundaries, regardless of the input data. The random pattern is simply defining the same boundary that would be occupied by one of the digits.

(ii)

If the input sensors are subjected to lots of noise (random signals), it is entirely reasonable to assume that the neural net would classify objects mistakenly. Clear evidence in the previous exercise. Intuitively, the neural net is attempting to discriminate fine details in the data, and ignores everything else. High noise make it difficult for the system to tell the difference (ignore). Is there a way for all nodes to be zero?

(iii)

Perhaps it would be possible to alter the training data to make the image boundary larger. ie if the net is looking for smaller details in the data, perhaps it could be widened to detect larger or more significant details to signal a strong detection. When images with high noise are processed, it could require more positive data identification before positive signaling. Basically, reduce the amount of signal the process ignores.