# HW5

*Yigao Li*

*February 23, 2018*

## 1. Feed Forward

### (a)

(See Figure 1.)

### (b)

$$h_1 = tanh(1.2 + 4.2x_1 - 0.5x_2)$$

### (c)

$$z = tanh(5 - 8h_1 + 1.5h_2)$$

### (d)

$$\begin{aligned} z &= tanh(5 - 8h_1 + 1.5h_2) \\ &= tanh(5 - 8tanh(1.2 + 4.2x_1 - 0.5x_2) + 1.5tanh(-30 + 20x_1 - 40x_2)) \end{aligned}$$

## 2. Multiple Solutions

Matrix of weights from input to hidden layer was $B = \begin{pmatrix} 0 & 5 & 0 \\ 0 & 0 & 7 \end{pmatrix}$, $w = \begin{pmatrix} -5 \\ 4 \\ 6 \end{pmatrix}$

Similar sets of weights lead to the same output:

1. $B = \begin{pmatrix} 0 & -5 & 0 \\ 0 & 0 & 7 \end{pmatrix}$, $w = \begin{pmatrix} -5 \\ -4 \\ 6 \end{pmatrix}$
2. $B = \begin{pmatrix} 0 & 5 & 0 \\ 0 & 0 & -7 \end{pmatrix}$, $w = \begin{pmatrix} -5 \\ 4 \\ -6 \end{pmatrix}$
3. $B = \begin{pmatrix} 0 & -5 & 0 \\ 0 & 0 & -7 \end{pmatrix}$, $w = \begin{pmatrix} -5 \\ -4 \\ -6 \end{pmatrix}$

Assume that $g(y) = tanh(y)$, $g$ is an odd function. Output $z = tanh(-5 + 4tanh5x_1 + 6tanh5x_2)$. Switching signs of coefficients of $x_1$ and $h_1$ does not change final output, similar to $x_2$ and $h_2$:

$$\begin{aligned} z &= tanh(-5 + 4tanh5x_1 + 6tanh5x_2) \\ &= tanh(-5 - 4tanh(-5x_1) - 6tanh(-5x_2)) \\ &= tanh(-5 - 4tanh(-5x_1) + 6tanh5x_2) \\ &= tanh(-5 + 4tanh5x_1 - 6tanh(-5x_2)) \end{aligned}$$

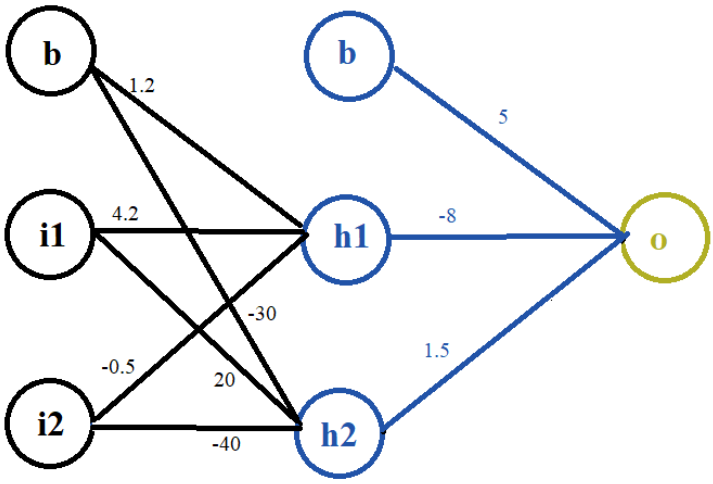



Figure 1: Diagram of given neural network architecture

## 3. ANNs and Multiple Linear Regression

### (a)

```
library(nnet)
```

```
## Warning: package 'nnet' was built under R version 3.4.3
```

```
set.seed(1234)
Advertising <- read.csv("Advertising.csv")
lin.model <- lm(Sales ~ TV + Radio + Newspaper, data = Advertising)
summary(lin.model)
```

```
##
## Call:
## lm(formula = Sales ~ TV + Radio + Newspaper, data = Advertising)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.8277 -0.8908  0.2418  1.1893  2.8292
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.938889   0.311908   9.422   <2e-16 ***
## TV           0.045765   0.001395  32.809   <2e-16 ***
## Radio        0.188530   0.008611  21.893   <2e-16 ***
## Newspaper   -0.001037   0.005871  -0.177     0.86
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.686 on 196 degrees of freedom
## Multiple R-squared:  0.8972, Adjusted R-squared:  0.8956
## F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

```
nnet.model <- nnet(Sales ~ TV + Radio + Newspaper, data = Advertising, size = 0, linout = TRUE,
                   skip = TRUE)
```

```
## # weights:  4
## initial  value 102721.171716
## final  value 556.825263
## converged
```

```
summary(nnet.model)
```

```
## a 3-0-1 network with 4 weights
## options were - skip-layer connections  linear output units
##  b->o i1->o i2->o i3->o
##  2.94  0.05  0.19  0.00
```

```
sum(residuals(lin.model)^2)
```

```
## [1] 556.8253
```

```
sum(residuals(nnet.model)^2)
```

```
## [1] 556.8253
```

Sum of squared errors are exactly the same for both methods.

## (b)

```
Advert2 <- as.data.frame(scale(Advertising))
head(Advertising)
```

```
##   X    TV Radio Newspaper Sales
## 1 1 230.1  37.8      69.2  22.1
## 2 2  44.5  39.3      45.1  10.4
## 3 3  17.2  45.9      69.3   9.3
## 4 4 151.5  41.3      58.5  18.5
## 5 5 180.8  10.8      58.4  12.9
## 6 6   8.7  48.9      75.0   7.2
```

```
head(Advert2)
```

```
##           X          TV      Radio Newspaper      Sales
## 1 -1.719098  0.96742460  0.9790656 1.7744925  1.5481681
## 2 -1.701821 -1.19437904  1.0800974 0.6679027 -0.6943038
## 3 -1.684543 -1.51235985  1.5246374 1.7790842 -0.9051345
## 4 -1.667266  0.05191939  1.2148065 1.2831850  0.8581768
## 5 -1.649989  0.39319551 -0.8395070 1.2785934 -0.2151431
## 6 -1.632711 -1.61136487  1.7267010 2.0408088 -1.3076295
```

```
newAd <- Advert2
newAd$Sales <- Advertising$Sales
head(newAd)
```

```
##           X          TV      Radio Newspaper Sales
## 1 -1.719098  0.96742460  0.9790656 1.7744925  22.1
## 2 -1.701821 -1.19437904  1.0800974 0.6679027  10.4
## 3 -1.684543 -1.51235985  1.5246374 1.7790842   9.3
## 4 -1.667266  0.05191939  1.2148065 1.2831850  18.5
## 5 -1.649989  0.39319551 -0.8395070 1.2785934  12.9
## 6 -1.632711 -1.61136487  1.7267010 2.0408088   7.2
```

All data are converted to gaussian scaled values.

### (c)

```
lin.model.scale <- lm(Sales ~ TV + Radio + Newspaper, data = newAd)
summary(lin.model.scale)
```

```
##
## Call:
## lm(formula = Sales ~ TV + Radio + Newspaper, data = newAd)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.8277 -0.8908  0.2418  1.1893  2.8292
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  14.0225     0.1192 117.655   <2e-16 ***
## TV            3.9291     0.1198  32.809   <2e-16 ***
## Radio         2.7991     0.1278  21.893   <2e-16 ***
## Newspaper    -0.0226     0.1279  -0.177     0.86
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.686 on 196 degrees of freedom
## Multiple R-squared:  0.8972, Adjusted R-squared:  0.8956
## F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

```
sum(residuals(lin.model.scale)^2)
```

```
## [1] 556.8253
```

## 4. MNIST Revisited

### (a)

```
load("mnist_all.RData")
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 3.4.3
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
index <- (train$y == 4 | train$y == 7)
df <- train$x[index,]
df.y <- train$y[index]
df <- as.data.frame(df)
df$y <- (df.y-4)/3  # 0 for number 4;   1 for number 7
a <- 711
b <- 298
var(df[,a])
```

```
## [1] 6673.933
```

```
var(df[,b])
```

```
## [1] 11220.55
```

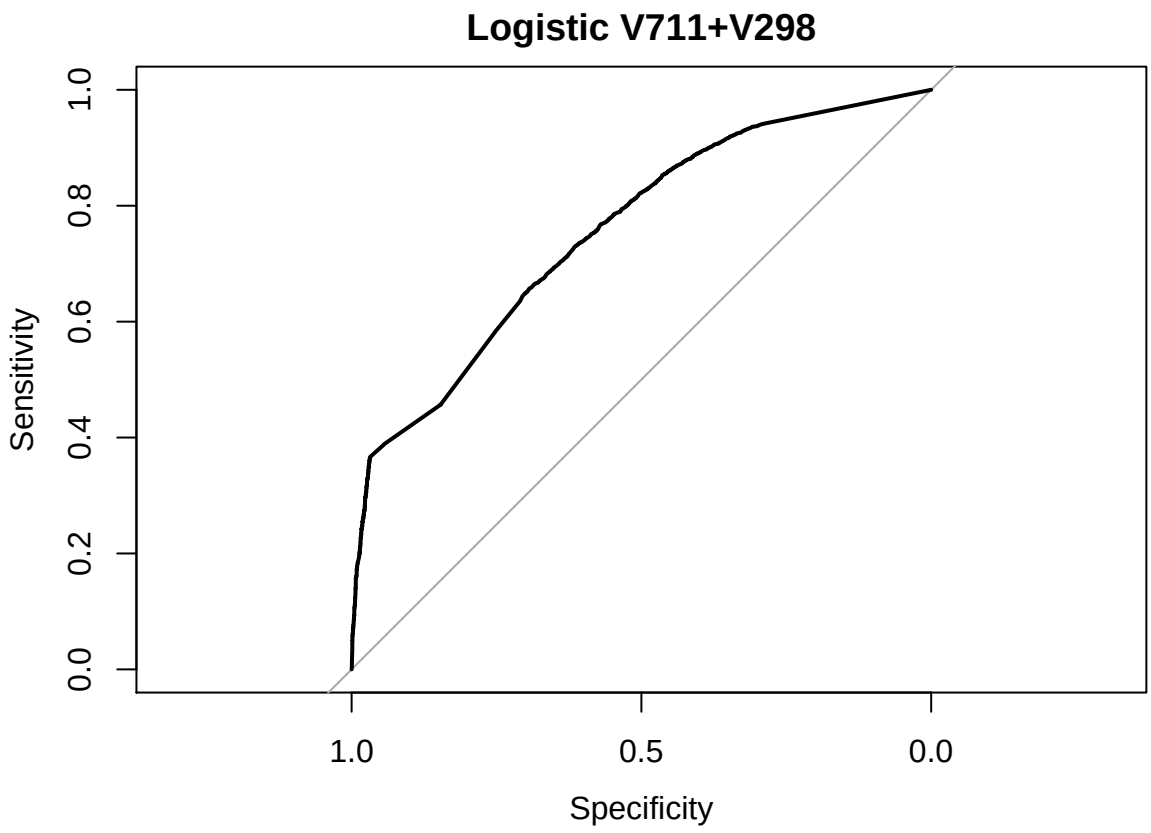
```
cor(df[,a], df[,b])
```

```
## [1] 0.009442966
```

```
model.1 <- glm(y ~ V711 + V298, data = df, family = binomial)
summary(model.1)
```

```
##
## Call:
## glm(formula = y ~ V711 + V298, family = binomial, data = df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9213  -0.9918   0.1721   1.0906   1.6796
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.1308598  0.0383015  -29.52   <2e-16 ***
## V711         0.0158225  0.0004971   31.83   <2e-16 ***
## V298         0.0052907  0.0001985   26.66   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 16769  on 12106  degrees of freedom
## Residual deviance: 13856  on 12104  degrees of freedom
## AIC: 13862
##
## Number of Fisher Scoring iterations: 5
```

```
df$pred <- predict(model.1, type = "response")
myroc <- roc(df$y, df$pred)
plot(myroc, main = "Logistic V711+V298")
```




```
auc(df$y, df$pred)
```

```
## Area under the curve: 0.7543
```

## (b)

```
set.seed(1)
model.2 <- nnet(y ~ V711 + V298, data = df, size = 1)
```

```
## # weights:  5
## initial  value 2993.047137
## iter  10 value 2474.687781
## iter  20 value 2288.905982
## iter  30 value 2288.030887
## iter  40 value 2287.453102
## iter  50 value 2287.240971
## iter  60 value 2287.074669
## iter  70 value 2286.934932
## iter  80 value 2286.853979
## iter  90 value 2286.808223
## final  value 2286.790916
## converged
```

```
summary(model.2)
```

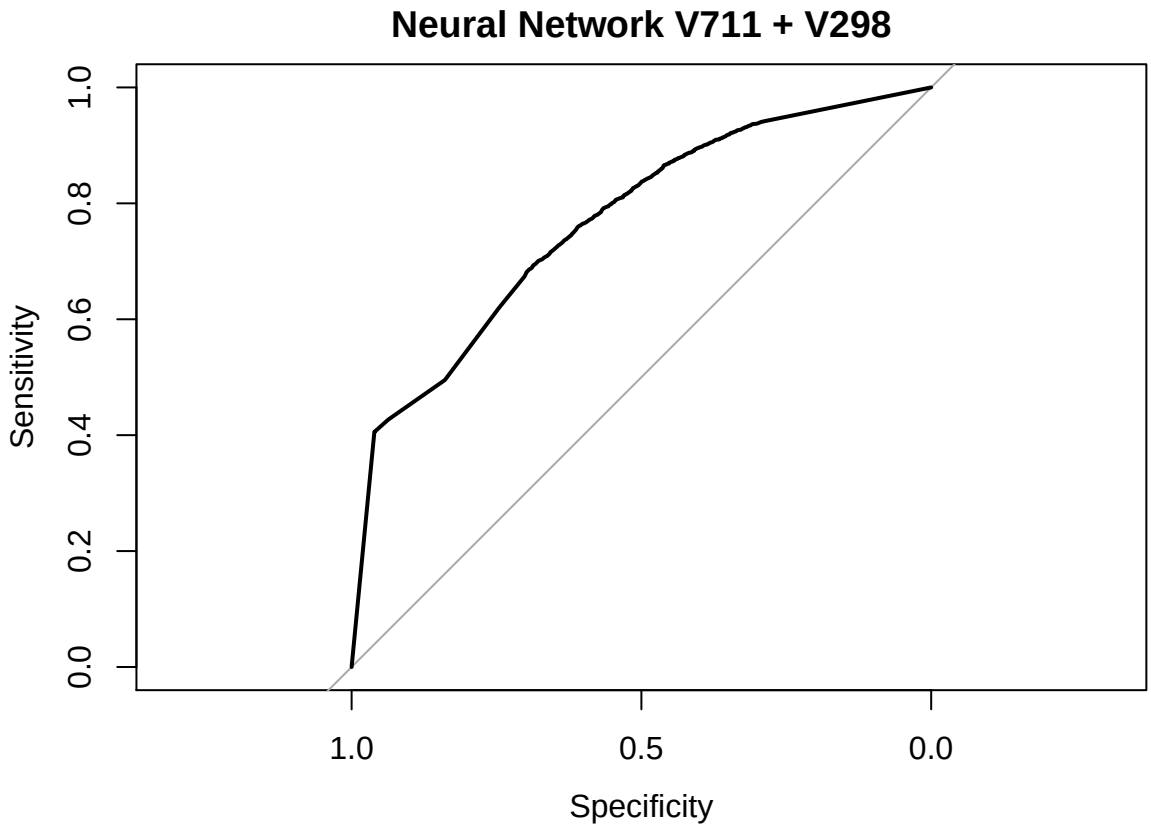
```
## a 2-1-1 network with 5 weights
```

```
## options were -
##  b->h1 i1->h1 i2->h1
##  -1.69  -8.02   0.00
##   b->o  h1->o
##   2.39 -23.65
```

```
df$pred <- predict(model.2, type = "raw")
myroc <- roc(df$y, df$pred)
```

```
## Warning in roc.default(df$y, df$pred): Deprecated use a matrix as
## predictor. Unexpected results may be produced, please pass a numeric
## vector.
```

```
plot(myroc, main = "Neural Network V711 + V298")
```




```
auc(df$y, df$pred)
```

```
## Warning in roc.default(response, predictor, auc = TRUE, ...): Deprecated
## use a matrix as predictor. Unexpected results may be produced, please pass
## a numeric vector.
```

```
## Area under the curve: 0.7663
```

Results from ANN slightly improves from logistic regression by 1.2% of training accuracy. 2 plots indicate that true positive rate is higher for ANN method with low false positive rate.

(c)

```
set.seed(2)

model.3 <- nnet(y ~ V711 + V298, data = df, size = 2)
```

```
## # weights:  9
## initial  value 2968.910960
## iter  10 value 2488.740396
## iter  20 value 2287.703570
## iter  30 value 2283.197239
## iter  40 value 2282.393015
## iter  50 value 2280.222002
## iter  60 value 2274.219847
## iter  70 value 2270.629453
## iter  80 value 2270.131971
## final  value 2269.942259
## converged
```

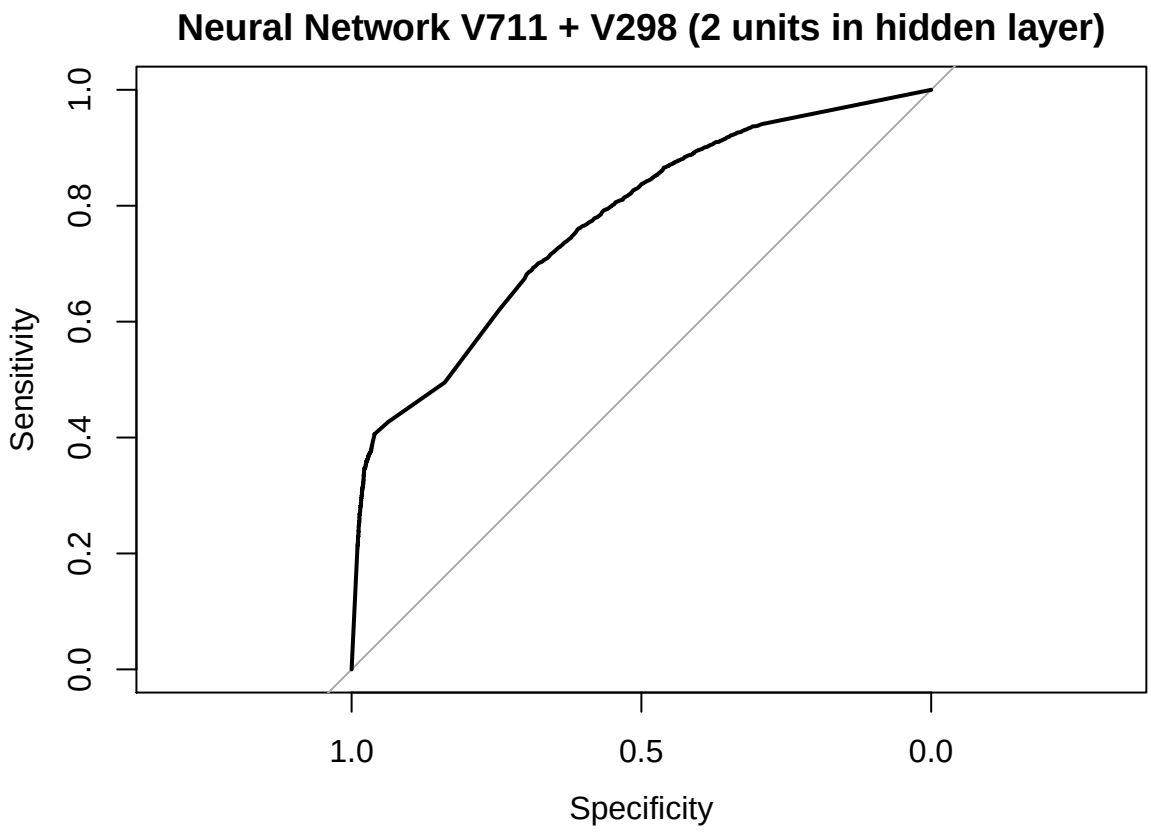
```
summary(model.3)
```

```
## a 2-2-1 network with 9 weights
## options were -
##  b->h1 i1->h1 i2->h1
##  -4.06   2.66   0.01
##  b->h2 i1->h2 i2->h2
##   2.01   0.00   0.07
##  b->o h1->o h2->o
## -8.93  3.28  8.31
```

```
df$pred <- predict(model.3, type = "raw")
myroc <- roc(df$y, df$pred)
```

```
## Warning in roc.default(df$y, df$pred): Deprecated use a matrix as
## predictor. Unexpected results may be produced, please pass a numeric
## vector.
```

```
plot(myroc, main = "Neural Network V711 + V298 (2 units in hidden layer)")
```




```
auc(df$y, df$pred)
```

```
## Warning in roc.default(response, predictor, auc = TRUE, ...): Deprecated
## use a matrix as predictor. Unexpected results may be produced, please pass
## a numeric vector.
```

```
## Area under the curve: 0.7694
```

```
model.4 <- nnet(y ~ V711 + V298, data = df, size = 4)
```

```
## # weights:  17
## initial  value 3060.513016
## iter  10 value 2373.051947
## iter  20 value 2306.585679
## iter  30 value 2278.954101
## iter  40 value 2276.525340
## iter  50 value 2274.768698
## iter  60 value 2270.848447
## iter  70 value 2270.235412
## iter  80 value 2269.882956
## iter  90 value 2269.521389
## iter 100 value 2269.080260
## final  value 2269.080260
## stopped after 100 iterations
```

```
summary(model.4)
```

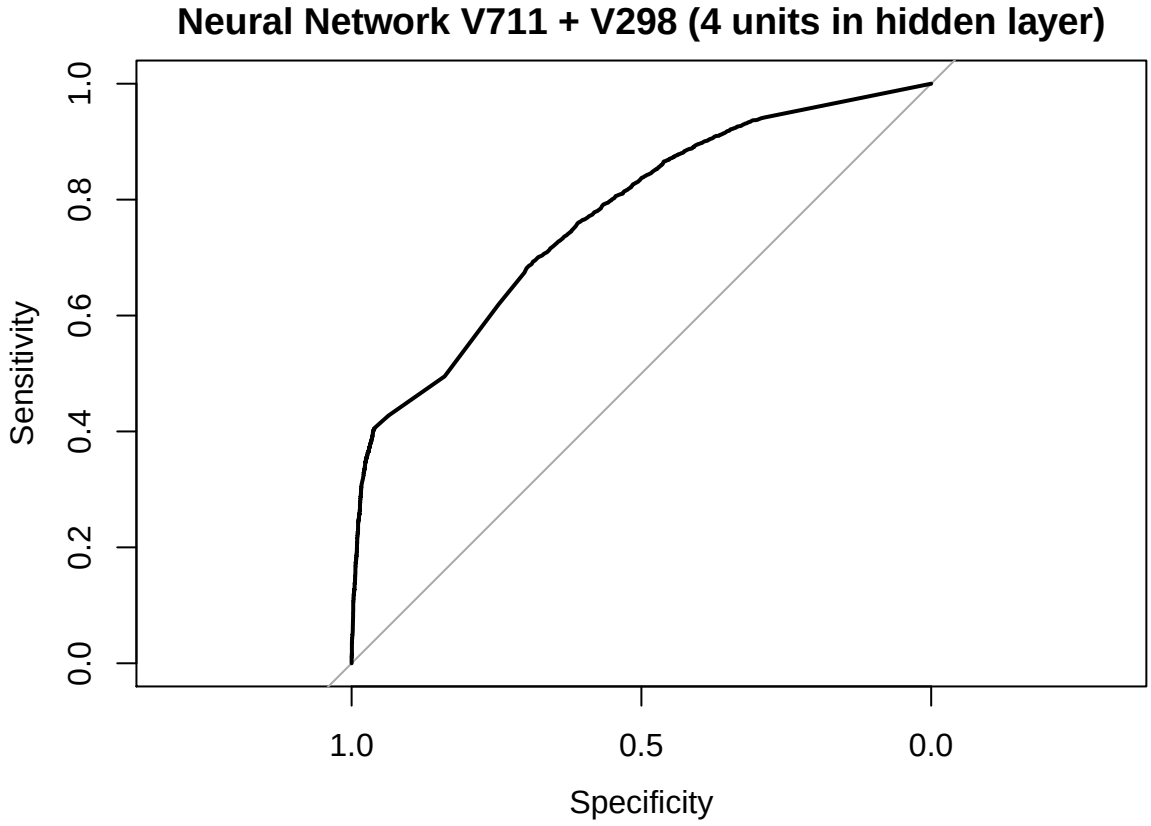
```
## a 2-4-1 network with 17 weights
```

```
## options were -
##  b->h1 i1->h1 i2->h1
##  -0.27   2.75  -1.51
##  b->h2 i1->h2 i2->h2
##   5.24  -0.02  -0.02
##  b->h3 i1->h3 i2->h3
##  -2.90   2.67  -0.01
##  b->h4 i1->h4 i2->h4
##  -0.42  -0.15  -0.02
##  b->o h1->o h2->o h3->o h4->o
##  1.47 -1.08 -1.90  2.90 -2.01
```

```
df$pred <- predict(model.4, type = "raw")
myroc <- roc(df$y, df$pred)
```

```
## Warning in roc.default(df$y, df$pred): Deprecated use a matrix as
## predictor. Unexpected results may be produced, please pass a numeric
## vector.
```

```
plot(myroc, main = "Neural Network V711 + V298 (4 units in hidden layer)")
```




```
auc(df$y, df$pred)
```

```
## Warning in roc.default(response, predictor, auc = TRUE, ...): Deprecated
## use a matrix as predictor. Unexpected results may be produced, please pass
## a numeric vector.
```

```
## Area under the curve: 0.7697
```

```r
model.5 <- nnet(y ~ V711 + V298, data = df, size = 6)
```

```
## # weights:  25
## initial  value 3728.023771
## iter  10 value 2337.455642
## iter  20 value 2276.452345
## iter  30 value 2269.746075
## iter  40 value 2268.582122
## iter  50 value 2267.717088
## iter  60 value 2267.244629
## iter  70 value 2266.949837
## iter  80 value 2266.799327
## iter  90 value 2266.750380
## iter 100 value 2266.609377
## final  value 2266.609377
## stopped after 100 iterations
```

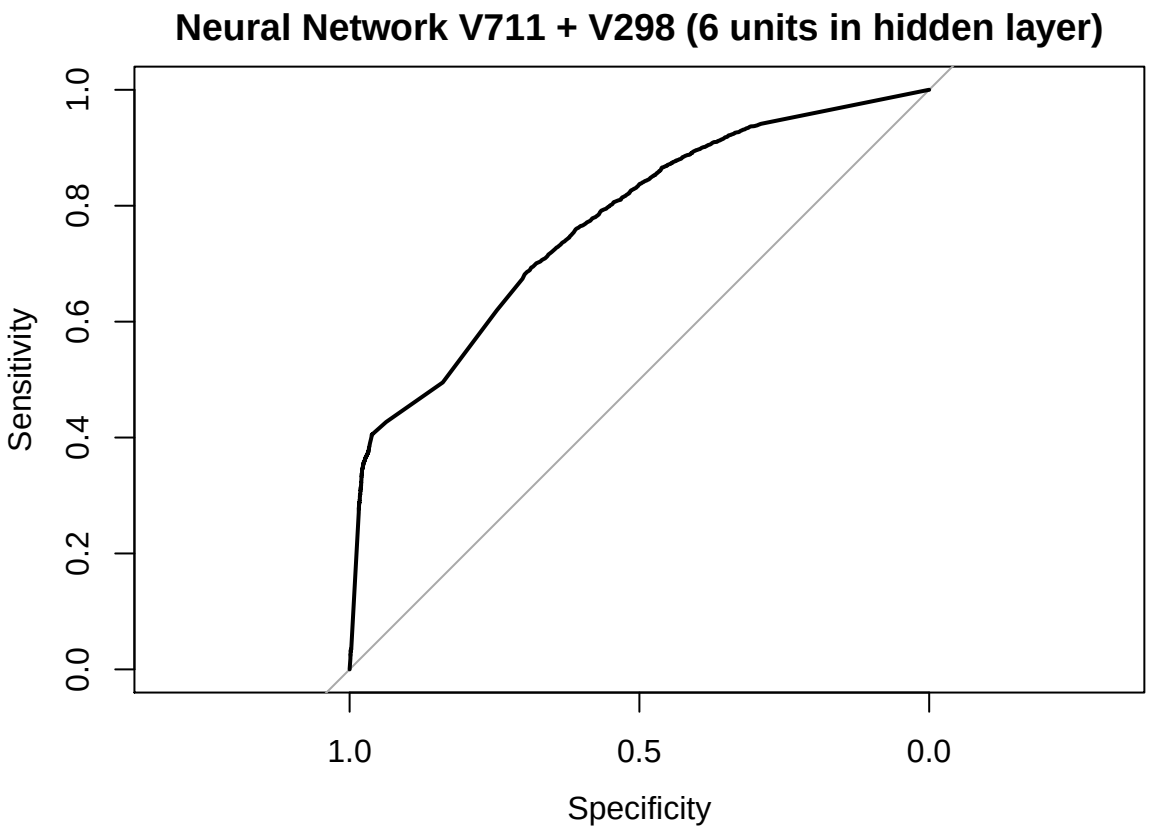
```r
summary(model.5)
```

```
## a 2-6-1 network with 25 weights
## options were -
##  b->h1 i1->h1 i2->h1
##  -0.99  -0.05   1.41
##  b->h2 i1->h2 i2->h2
##   0.75  -0.26  -0.23
##  b->h3 i1->h3 i2->h3
##  -6.19   2.21   5.23
##  b->h4 i1->h4 i2->h4
## -10.52   6.24   0.03
##  b->h5 i1->h5 i2->h5
##   7.86  -0.73   2.70
##  b->h6 i1->h6 i2->h6
##  -0.22  -0.17  -0.01
##   b->o  h1->o  h2->o  h3->o  h4->o  h5->o  h6->o
##   0.12   2.55  -0.61  -1.65   2.85  -1.05  -1.95
```

```r
df$pred <- predict(model.5, type = "raw")
myroc <- roc(df$y, df$pred)
```

```
## Warning in roc.default(df$y, df$pred): Deprecated use a matrix as
## predictor. Unexpected results may be produced, please pass a numeric
## vector.
```

```r
plot(myroc, main = "Neural Network V711 + V298 (6 units in hidden layer)")
```




```
auc(df$y, df$pred)
```

```
## Warning in roc.default(response, predictor, auc = TRUE, ...): Deprecated
## use a matrix as predictor. Unexpected results may be produced, please pass
## a numeric vector.
```

```
## Area under the curve: 0.769
```

The results did not improve. It is overfitting.