HW 13

Leah Fawzi

12/10/2018

```
library(nnet)
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

I'm looking at the first 1000 entries in the mnist train data set and changing the sample labels so that each sample is labeled as a 0 if the image is not a 3 and as a 1 if the image is a 3.

```
mtrain <- read.csv("mnist_train.csv", header=F) %>% as.matrix
train_classification <- mtrain[,1]

mtrain<- mtrain[,-1]/256
x <- mtrain[1:1000,]
y <- factor(train_classification[1:1000], levels=0:10,labels=c(0,0,0,1,0,0,0,0,0,0,0)) %>% factor
colnames(x)<- 1:784
```

```
print(head(train_classification))
```

```
## [1] 5 0 4 1 9 2
```

```
print(head(y))
```

```
## [1] 0 0 0 0 0 0
## Levels: 0 1
```

First, I will fit the data to a neural net with decay of 0 and a range of sizes and I'm using cross validation to find the most optimal size. Because of time constraints, the range of sizes is not as big as I'd have liked.

```
fitControl <- trainControl(
  method = "repeatedcv",
  number = 2,
  repeats = 2)

tuning_df <- data.frame(size=9:12, decay=0)

t_out <- caret::train(x=x, y=y, method="nnet",
                      trControl = fitControl,
                      tuneGrid=tuning_df, maxit=1000, MaxNWts=100000)
```

```
## # weights:  7075
## initial  value 414.662041
## iter  10 value 15.050118
## iter  20 value 0.699983
## iter  30 value 0.033776
## iter  40 value 0.004101
## iter  50 value 0.000672
## iter  60 value 0.000220
## iter  70 value 0.000127
## final  value 0.000079
## converged
## # weights:  7861
## initial  value 300.150861
## iter  10 value 13.161851
## iter  20 value 0.099581
## iter  30 value 0.000738
## iter  40 value 0.000224
## final  value 0.000092
## converged
## # weights:  8647
## initial  value 270.989905
## iter  10 value 39.089937
## iter  20 value 3.350639
## iter  30 value 0.070434
## iter  40 value 0.003235
## iter  50 value 0.000333
## iter  60 value 0.000178
## iter  70 value 0.000138
## iter  80 value 0.000130
## iter  90 value 0.000101
## iter  90 value 0.000100
## iter  90 value 0.000100
## final  value 0.000100
## converged
## # weights:  9433
## initial  value 522.999060
## iter  10 value 14.559489
## iter  20 value 0.454795
## iter  30 value 0.007898
## iter  40 value 0.001341
## iter  50 value 0.000110
## iter  50 value 0.000075
## iter  50 value 0.000075
## final  value 0.000075
## converged
## # weights:  7075
## initial  value 403.420999
## iter  10 value 40.305265
## iter  20 value 17.693941
## iter  30 value 11.969095
## iter  40 value 8.346466
## iter  50 value 7.165371
## iter  60 value 7.119215
## iter  70 value 7.115333
## iter  80 value 7.114343
## iter  90 value 7.062137
## iter 100 value 6.300299
## iter 110 value 0.458988
## iter 120 value 0.021898
## iter 130 value 0.003837
## iter 140 value 0.002312
## iter 150 value 0.001142
## iter 160 value 0.000490
## iter 170 value 0.000489
## iter 180 value 0.000489
## final  value 0.000489
## converged
## # weights:  7861
## initial  value 441.566587
## iter  10 value 61.718204
## iter  20 value 21.363389
## iter  30 value 5.210099
## iter  40 value 1.440485
## iter  50 value 0.024248
## iter  60 value 0.009232
## iter  70 value 0.004256
## iter  80 value 0.000539
## iter  90 value 0.000141
## final  value 0.000094
## converged
## # weights:  8647
## initial  value 338.667746
## iter  10 value 14.310091
## iter  20 value 0.221664
## iter  30 value 0.007376
## iter  40 value 0.000360
## iter  50 value 0.000162
## final  value 0.000094
## converged
## # weights:  9433
## initial  value 418.617928
## iter  10 value 155.601712
## iter  20 value 144.336088
## iter  30 value 73.684763
## iter  40 value 51.265991
## iter  50 value 45.291155
## iter  60 value 45.037467
## iter  70 value 44.817401
## iter  80 value 28.598211
## iter  90 value 3.285218
## iter 100 value 3.154277
## iter 110 value 3.141694
## iter 120 value 3.139599
## final  value 3.139492
## converged
## # weights:  7075
## initial  value 609.003386
## iter  10 value 53.611236
## iter  20 value 9.691984
## iter  30 value 0.129835
## iter  40 value 0.007094
## iter  50 value 0.000747
## iter  60 value 0.000304
## iter  70 value 0.000106
## iter  70 value 0.000097
## iter  70 value 0.000097
## final  value 0.000097
## converged
## # weights:  7861
## initial  value 330.360179
## iter  10 value 33.421148
## iter  20 value 15.186412
## iter  30 value 5.210196
## iter  40 value 1.972296
## iter  50 value 0.047356
## iter  60 value 0.002870
## final  value 0.000034
## converged
## # weights:  8647
## initial  value 580.956151
## iter  10 value 18.414023
## iter  20 value 0.477556
## iter  30 value 0.008813
## iter  40 value 0.000863
## iter  50 value 0.000260
## iter  60 value 0.000156
## final  value 0.000078
## converged
## # weights:  9433
## initial  value 516.314830
## iter  10 value 153.473832
## iter  10 value 153.473832
## iter  10 value 153.473832
## final  value 153.473832
## converged
## # weights:  7075
## initial  value 636.462355
## iter  10 value 37.488379
## iter  20 value 0.251042
## iter  30 value 0.011015
## iter  40 value 0.000430
## final  value 0.000077
## converged
## # weights:  7861
## initial  value 473.163218
## iter  10 value 30.451471
## iter  20 value 0.434303
## iter  30 value 0.009195
## iter  40 value 0.001246
## iter  50 value 0.000182
## final  value 0.000098
## converged
## # weights:  8647
## initial  value 368.895230
## iter  10 value 22.277951
## iter  20 value 0.564992
## iter  30 value 0.074108
## iter  40 value 0.018841
## iter  50 value 0.004513
## iter  60 value 0.000822
## iter  70 value 0.000137
## iter  70 value 0.000074
## iter  70 value 0.000074
## final  value 0.000074
## converged
## # weights:  9433
## initial  value 993.988034
## iter  10 value 19.518280
## iter  20 value 0.138556
## iter  30 value 0.001383
## iter  40 value 0.000318
## final  value 0.000059
## converged
## # weights:  7075
## initial  value 491.871821
## iter  10 value 131.191806
## iter  20 value 60.199831
## iter  30 value 35.054842
## iter  40 value 22.549239
## iter  50 value 19.157419
## iter  60 value 17.899822
## iter  70 value 17.775744
## iter  80 value 16.494233
## iter  90 value 13.455168
## iter 100 value 11.626561
## iter 110 value 10.953351
## iter 120 value 10.948514
## iter 130 value 6.020477
## iter 140 value 5.052738
## iter 150 value 5.050741
## iter 160 value 5.050281
## iter 170 value 5.048848
## iter 180 value 4.585025
## iter 190 value 4.526105
## iter 200 value 4.526042
## iter 210 value 1.562345
## iter 220 value 1.395597
## iter 230 value 1.388220
## iter 240 value 1.386838
## final  value 1.386747
## converged
```

```r
print(t_out)
```

```
## Neural Network
```

```
##
## 1000 samples
##  784 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (2 fold, repeated 2 times)
## Summary of sample sizes: 499, 501, 499, 501
## Resampling results across tuning parameters:
##
##   size  Accuracy   Kappa
##    9    0.9659979  0.7888281
##   10    0.9560058  0.7304931
##   11    0.9629999  0.7711268
##   12    0.9480278  0.5732687
##
## Tuning parameter 'decay' was held constant at a value of 0
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were size = 9 and decay = 0.
```

In my script I got size=11
so that is the size I use in
the following fit. I used
cache=F in Rmarkdown
but it doesn't seem to be working
- I am unable to run it again
to add this into my commentary.

So, the optimal size is 11 so we will use that size as we test what decay is optimal.

```r
tuning_df2 <- data.frame(size=11, decay=c(0,0.5,1,2))
t_out2 <- caret::train(x=x, y=y, method="nnet",
                      trControl = fitControl,
                      tuneGrid=tuning_df2, maxit=1000, MaxNWts=100000)
```

```
## # weights:  8647
## initial  value 257.671630
## iter  10 value 22.443630
## iter  20 value 0.074892
## iter  30 value 0.001267
## iter  40 value 0.000252
## final  value 0.000086
## converged
## # weights:  8647
## initial  value 1269.043524
## iter  10 value 265.966043
## iter  20 value 71.718693
## iter  30 value 44.029028
## iter  40 value 38.114954
## iter  50 value 33.338615
## iter  60 value 30.660052
## iter  70 value 29.370835
## iter  80 value 29.131784
## iter  90 value 29.007648
## iter 100 value 28.929597
## iter 110 value 28.912299
## iter 120 value 28.907344
## iter 130 value 28.903648
## iter 140 value 28.902216
## iter 150 value 28.902076
## final  value 28.902073
## converged
## # weights:  8647
## initial  value 1628.581412
## iter  10 value 331.471785
## iter  20 value 88.502899
## iter  30 value 55.366693
## iter  40 value 49.681042
## iter  50 value 47.655228
## iter  60 value 46.616543
## iter  70 value 46.385640
## iter  80 value 46.218796
## iter  90 value 46.161147
## iter 100 value 46.152497
## iter 110 value 46.144820
## iter 120 value 46.131596
## iter 130 value 46.099259
## iter 140 value 45.959178
## iter 150 value 45.886223
## iter 160 value 45.780283
## iter 170 value 45.751347
## iter 180 value 45.735678
## iter 190 value 45.732359
## iter 200 value 45.731435
## iter 210 value 45.731202
## iter 220 value 45.730941
## iter 230 value 45.730808
## iter 240 value 45.730674
## final  value 45.730671
## converged
## # weights:  8647
## initial  value 3214.884358
## iter  10 value 193.691680
## iter  20 value 105.325488
## iter  30 value 89.712124
## iter  40 value 80.020810
## iter  50 value 72.431831
## iter  60 value 70.878726
## iter  70 value 69.957210
## iter  80 value 69.514401
## iter  90 value 69.391065
## iter 100 value 69.381406
## iter 110 value 69.378488
## iter 120 value 69.377993
## final  value 69.377956
## converged
## # weights:  8647
## initial  value 319.658256
## iter  10 value 43.163087
## iter  20 value 17.857827
## iter  30 value 4.580018
## iter  40 value 0.473906
## iter  50 value 0.017526
## iter  60 value 0.001264
## iter  70 value 0.000369
## iter  80 value 0.000251
## iter  90 value 0.000121
## final  value 0.000091
## converged
## # weights:  8647
## initial  value 1274.302541
## iter  10 value 237.687845
## iter  20 value 77.079922
## iter  30 value 49.482729
## iter  40 value 43.145352
## iter  50 value 35.458146
## iter  60 value 30.909357
## iter  70 value 30.043720
## iter  80 value 29.852540
## iter  90 value 29.735179
## iter 100 value 29.690969
## iter 110 value 29.658354
## iter 120 value 29.635898
## iter 130 value 29.616441
## iter 140 value 29.605144
## iter 150 value 29.598232
## iter 160 value 29.590520
## iter 170 value 29.589011
## iter 180 value 29.588375
## iter 190 value 29.588192
## final  value 29.588183
## converged
## # weights:  8647
## initial  value 1833.980847
## iter  10 value 204.728900
## iter  20 value 118.256734
## iter  30 value 73.667221
## iter  40 value 59.185469
## iter  50 value 52.316034
## iter  60 value 48.574595
## iter  70 value 47.700889
## iter  80 value 46.897786
## iter  90 value 46.200704
## iter 100 value 46.111071
## iter 110 value 46.053682
## iter 120 value 46.004096
## iter 130 value 45.990697
## iter 140 value 45.974341
## iter 150 value 45.960066
## iter 160 value 45.951638
## iter 170 value 45.947559
## iter 180 value 45.946327
## final  value 45.946286
## converged
## # weights:  8647
## initial  value 3418.798750
## iter  10 value 250.535965
## iter  20 value 133.700706
## iter  30 value 92.139919
## iter  40 value 81.242764
## iter  50 value 74.742246
## iter  60 value 71.148739
## iter  70 value 70.546225
## iter  80 value 70.441547
## iter  90 value 70.254759
## iter 100 value 70.015925
## iter 110 value 69.825754
## iter 120 value 69.763426
## iter 130 value 69.746313
## iter 140 value 69.730439
## iter 150 value 69.708177
## iter 160 value 69.707046
## final  value 69.707034
## converged
## # weights:  8647
## initial  value 254.385193
## iter  10 value 13.581400
## iter  20 value 0.100993
## iter  30 value 0.006860
## iter  40 value 0.001157
## iter  50 value 0.000184
## iter  60 value 0.000101
## iter  60 value 0.000069
## iter  60 value 0.000069
## final  value 0.000069
## converged
## # weights:  8647
## initial  value 1025.608418
## iter  10 value 300.065395
## iter  20 value 61.820756
## iter  30 value 37.975921
## iter  40 value 31.241375
## iter  50 value 29.626874
## iter  60 value 29.404050
## iter  70 value 29.329231
## iter  80 value 29.300133
## iter  90 value 29.283294
## iter 100 value 29.263906
## iter 110 value 29.252840
## iter 120 value 29.242643
## iter 130 value 29.240795
## iter 140 value 29.240382
## iter 150 value 29.239873
## iter 160 value 29.239566
## iter 170 value 29.239463
## iter 180 value 29.238897
## iter 190 value 29.238858
## final  value 29.238857
## converged
## # weights:  8647
## initial  value 1742.970329
## iter  10 value 239.929994
## iter  20 value 90.009292
## iter  30 value 59.631541
## iter  40 value 50.747325
## iter  50 value 47.986248
## iter  60 value 46.771373
## iter  70 value 46.327404
## iter  80 value 46.193420
## iter  90 value 46.167703
## iter 100 value 46.134228
## iter 110 value 46.108584
## iter 120 value 46.105403
## iter 130 value 46.104413
## iter 140 value 46.104148
## final  value 46.104135
## converged
## # weights:  8647
## initial  value 3146.611283
## iter  10 value 296.433200
## iter  20 value 130.544865
## iter  30 value 86.998743
## iter  40 value 77.528379
## iter  50 value 72.508884
## iter  60 value 69.992465
## iter  70 value 69.525884
## iter  80 value 69.312193
## iter  90 value 69.226033
## iter 100 value 69.208689
## iter 110 value 69.203708
## iter 120 value 69.203184
## final  value 69.203130
## converged
## # weights:  8647
## initial  value 209.574284
## iter  10 value 8.409571
## iter  20 value 0.028427
## iter  30 value 0.001915
## iter  40 value 0.000672
## iter  50 value 0.000262
## iter  60 value 0.000209
## iter  70 value 0.000196
## iter  80 value 0.000182
## final  value 0.000081
## converged
## # weights:  8647
## initial  value 975.870070
## iter  10 value 224.923102
## iter  20 value 96.738356
## iter  30 value 41.129164
## iter  40 value 30.027001
## iter  50 value 28.593392
## iter  60 value 28.087540
## iter  70 value 27.863808
## iter  80 value 27.710376
## iter  90 value 27.604304
## iter 100 value 27.534142
## iter 110 value 27.501433
## iter 120 value 27.481691
## iter 130 value 27.467093
## iter 140 value 27.453879
## iter 150 value 27.441861
## iter 160 value 27.432333
## iter 170 value 27.431517
## final  value 27.431491
## converged
## # weights:  8647
## initial  value 1699.586988
## iter  10 value 187.614901
## iter  20 value 61.446507
## iter  30 value 46.874480
## iter  40 value 45.746707
## iter  50 value 44.680862
## iter  60 value 44.348810
## iter  70 value 44.247118
## iter  80 value 44.217322
## iter  90 value 44.183704
## iter 100 value 44.170101
## iter 110 value 44.169551
## final  value 44.169545
## converged
## # weights:  8647
## initial  value 3144.825928
## iter  10 value 299.088329
## iter  20 value 104.214720
## iter  30 value 76.083113
## iter  40 value 71.935549
## iter  50 value 69.349917
## iter  60 value 68.182644
## iter  70 value 67.980143
## iter  80 value 67.877828
## iter  90 value 67.867851
## iter 100 value 67.866427
## iter 110 value 67.866039
## final  value 67.866019
## converged
## # weights:  8647
## initial  value 1901.958167
## iter  10 value 299.020379
## iter  20 value 102.577634
## iter  30 value 73.892840
## iter  40 value 57.825126
## iter  50 value 48.616367
## iter  60 value 45.127266
## iter  70 value 43.908943
## iter  80 value 43.407206
## iter  90 value 43.131923
## iter 100 value 43.080812
## iter 110 value 43.042082
## iter 120 value 43.001572
## iter 130 value 42.957132
## iter 140 value 42.895471
## iter 150 value 42.856741
## iter 160 value 42.818231
## iter 170 value 42.792648
## iter 180 value 42.780109
## iter 190 value 42.738577
## iter 200 value 42.718637
## iter 210 value 42.714348
## iter 220 value 42.711498
## iter 230 value 42.710659
## iter 240 value 42.710578
## iter 250 value 42.710491
## iter 260 value 42.710450
## iter 260 value 42.710450
## iter 260 value 42.710450
## final  value 42.710450
## converged
```

```
print(t_out2)
```

```
## Neural Network
##
## 1000 samples
##  784 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (2 fold, repeated 2 times)
## Summary of sample sizes: 499, 501, 500, 500
## Resampling results across tuning parameters:
##
##   decay  Accuracy   Kappa
##   0.0    0.9644999  0.7761463
##   0.5    0.9715039  0.8192727
##   1.0    0.9705049  0.8070722
##   2.0    0.9690019  0.7905978
##
## Tuning parameter 'size' was held constant at a value of 11
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were size = 11 and decay = 0.5.
```

The optimal decay was 1. Again, the results were different from what I previously got.

Now we will check the accuracy of the two neural nets by computing the prediction error against the first 10,000 entries in the test dataset.

```
mtest <- read.csv("mnist_test.csv",header=F) %>% as.matrix
x2 <- mtest

train_classification2 <- mtest[,1]
mtest<- mtest[,-1]/256 #x matrix
y2 <- factor(train_classification2, levels=0:10,labels=c(0,0,0,1,0,0,0,0,0,0,0)) %>% factor
colnames(mtest)<- 1:784

true_y <- y2
pred_y1 <- predict(t_out, newdata = mtest)
pred_y2 <- predict(t_out2, newdata = mtest)
```

```r
n_samples <- nrow(x2)
error1 <- sum(true_y != pred_y1)/n_samples
cat("test prediction error with t_out", error1, "\n")
```

```
## test prediction error with t_out 0.0481
```

```r
error2 <- sum(true_y != pred_y2)/n_samples
cat("test prediction error with t_out2", error1, "\n")
```

```
## test prediction error with t_out2 0.0481
```