Walkthrough 8.2 - Neural Network Models

Code

We will be using the neuralnet package, so install that if you have not already:

Hide

```
install.packages("neuralnet")
```

```
Error in install.packages : Updating loaded packages
```

We will use the dataset prepared in walkthrough 8. Load the libraries needed for that as well as the code preparing the data for the model. However a wrinkle is that the neural network model function requires that our variables all be coded as numerical - i.e. we are not allowed to use factors.

Hide

```
df <-
  df %>%
  mutate_if(is.character, as.factor)
```

```
mutate_if: converted 'subject' from character to factor (0 new NA)
           converted 'enrollment_reason' from character to factor (0 new NA)
           converted 'semester' from character to factor (0 new NA)
```

Okay so that's the dataframe we so far from Walkthrough 8; now we need to adjust the variables that are non-numeric to be numeric. The syntax is basically identical to what we just did to turn character variables into factors:

Hide

```
df <-
  df %>%
  mutate_if(is.factor, as.numeric)
```

```
mutate_if: converted 'subject' from factor to double (0 new NA)
           converted 'enrollment_reason' from factor to double (0 new NA)
           converted 'semester' from factor to double (0 new NA)
```

Use glimpse to check that everything is some kind of numeric:

Hide

```
glimpse(df)
```

```
Rows: 464
Columns: 13
$ int               <dbl> 5.0, 4.2, 5.0, 5.0, 3.8, 5.0, 3.0, 4.2, 4.4, 3.4, 4.7, 4.0,
4.2, 4.2, 3.8, 4.0, 4.3, 4.3, 4…
$ uv                <dbl> 4.333333, 4.000000, 3.666667, 5.000000, 3.500000, 5.000000,
3.333333, 2.666667, 5.000000, 2…
$ pc                <dbl> 4.50, 3.50, 4.00, 3.50, 3.50, 3.50, 3.00, 3.00, 4.00, 3.00,
4.50, 3.00, 4.00, 3.50, 3.00, 4…
$ time_spent        <dbl> 1555.1667, 1382.7001, 860.4335, 1598.6166, 1481.8000, 1321.
8164, 1390.2167, 1479.4166, 2625…
$ final_grade       <dbl> 93.45372, 81.70184, 88.48758, 81.85260, 84.00000, 83.58827,
97.77778, 96.11872, 93.90452, 9…
$ subject           <dbl> 3, 4, 3, 4, 5, 1, 5, 3, 4, 4, 4, 5, 4, 3, 5, 1, 1, 3, 3, 5,
4, 1, 4, 1, 3, 3, 4, 5, 5, 5, 3…
$ enrollment_reason <dbl> 1, 1, 4, 1, 3, 4, 3, 1, 4, 1, 3, 3, 1, 3, 1, 5, 3, 3, 1, 1,
3, 4, 1, 1, 1, 1, 3, 1, 1, 1, 1…
$ semester          <dbl> 2, 1, 2, 2, 1, 2, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 2, 1, 1,
1, 2, 1, 1, 1, 1, 2, 1, 1, 1, 3…
$ cogproc           <dbl> 15.069737, 7.106667, 15.165854, 14.508000, 16.692000, 14.92
8571, 15.121111, 18.228710, 12.7…
$ social            <dbl> 6.200526, 6.140000, 5.052927, 6.133000, 7.534000, 7.368095,
5.693333, 4.918065, 7.981111, 6…
$ posemo            <dbl> 3.555526, 19.010000, 2.906098, 5.591000, 3.794000, 5.68666
7, 5.143889, 6.232581, 6.328889, …
$ negemo            <dbl> 1.1363158, 0.0000000, 1.4187805, 1.1520000, 1.2820000, 0.56
19048, 0.4194444, 1.1087097, 0.8…
$ n                 <dbl> 38, 3, 41, 10, 10, 21, 18, 31, 18, 12, 3, 16, 7, 42, 8, 24,
23, 23, 32, 18, 15, 21, 22, 24,…
```

There is an inherent danger in converting Factor and particularly Character features into numeric features. Numeric features contain both an ordering and a notion of relative distance between the values. For a dataset with a large number of factor variables we should lean more towards models built with decision trees that will not have these non-features like order and relative distance builtin. Or we should use a pivot type command to replace a column with multiple factors with multiple columns with just two values.

Neural networks are sensitive to the relative scales of the inputs. I do not completely understand why, but my instinct is that they can be quickly overwhelmed by a large scale variable and will then be unable to find the appropriate minimum. We can normalize our variables, in practice you would want to keep track of this transformation so you could apply the same transformation to new data. There are also different normalizations to try.

Inspect df if you like to see the change.

Now we divide our dataset into testing and training sets:

Hide

```
df_test <-
  df_test %>%
  select(-temp_id)
```

```
select: dropped one variable (temp_id)
```

You should have two dataframes one for training our model and then one for testing the final model we select.

Load the neuralnet pakcage:

Hide

```
library(neuralnet)
```

Then the syntax is nearly identical to what we used for a Random Forest, except the parameters now refer to the structure of the neural network and its fitting features.

Type

Hide

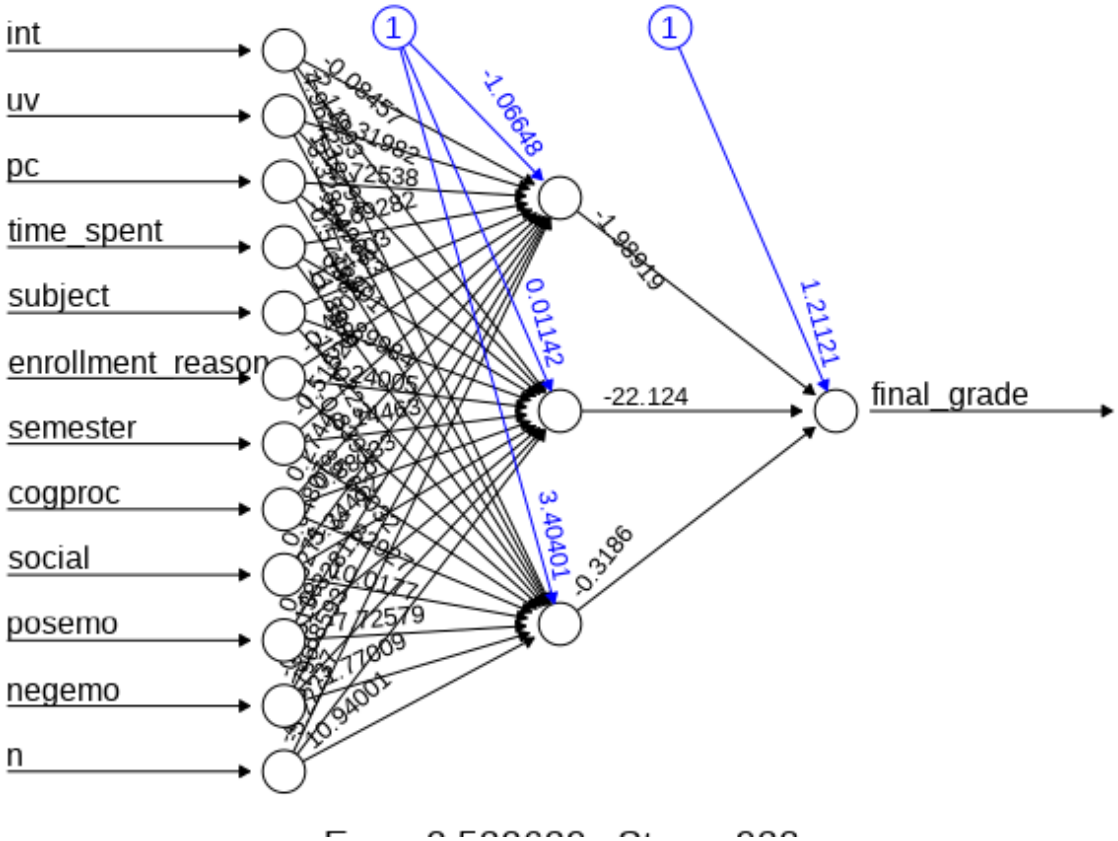
```
nn$result.matrix
```

```
                                    [,1]
error                          2.538638774
reached.threshold              0.009164512
steps                        938.000000000
Intercept.to.1layhid1         -1.066475900
int.to.1layhid1               -0.084573904
uv.to.1layhid1                -0.319820428
pc.to.1layhid1                 1.725379156
time_spent.to.1layhid1        -4.692818324
subject.to.1layhid1            0.143031169
enrollment_reason.to.1layhid1 -0.470407792
semester.to.1layhid1          -0.515261100
cogproc.to.1layhid1            0.274402655
social.to.1layhid1             0.048074767
posemo.to.1layhid1             0.402706058
negemo.to.1layhid1            -0.951378363
n.to.1layhid1                 -4.867962104
Intercept.to.1layhid2          0.011418375
int.to.1layhid2               -2.119930667
uv.to.1layhid2                 1.127937294
pc.to.1layhid2                -5.490834263
time_spent.to.1layhid2         0.880102903
subject.to.1layhid2           -0.899843584
enrollment_reason.to.1layhid2  1.240054638
semester.to.1layhid2          -0.144630841
cogproc.to.1layhid2            0.881328459
social.to.1layhid2            -5.344116968
posemo.to.1layhid2            -0.328108356
negemo.to.1layhid2            -2.685932782
n.to.1layhid2                 -5.222368687
Intercept.to.1layhid3          3.404008707
int.to.1layhid3                4.964226239
uv.to.1layhid3                -3.328040837
pc.to.1layhid3                -0.574882555
time_spent.to.1layhid3        11.898903362
subject.to.1layhid3          -12.022296331
enrollment_reason.to.1layhid3 -0.839350933
semester.to.1layhid3          85.660320027
cogproc.to.1layhid3           -3.829273998
social.to.1layhid3            10.017695343
posemo.to.1layhid3            -7.725787691
negemo.to.1layhid3            -1.770085254
n.to.1layhid3                 10.940006209
Intercept.to.final_grade       1.211211201
1layhid1.to.final_grade       -1.989186551
1layhid2.to.final_grade      -22.123996692
1layhid3.to.final_grade       -0.318602673
```

to inspect what we have. What is reported are the coefficients or weights assigned to each edge between nodes. In this case a single hidden layer with 3 nodes.

linear.output=TRUE you would change to FALSE if you were doing classification instead of regression.

R has a brilliant plot function for neural networks:



Although granted the main thing we usually learn his how impossible a task interpreting this type of model is. Note the extent to which the inputs are shuffled and overlap.

Let's check how it did on the testing set:

actual <dbl>	prediction <dbl>
0.93871751	0.8260598
0.65658904	0.4789402
0.73643780	0.8303325
0.80630822	0.8762320
0.76851380	0.8430358
0.94763665	0.8778464
0.83142731	0.8896876
0.98638316	0.8752510
0.91376004	0.8722150
0.87175506	0.8143059

1-10 of 92 rows Previous **1** 2 3 4 5 6 … 10 Next

We can write a little function that checks how we did:

Hide

```
accuracy(results)
```

```
[1] 0.9194169
```

The power is that neural networks have almost unlimited flexibility provided. We can increase both the number of layers and the number of nodes in each layer, and these are the primary parameters we tune for an individual problem:

Hide

```
nn2 = neuralnet(final_grade ~ ., data=df_train, hidden = c(5, 5), act.fct = "logisti
c", linear.output = FALSE)
Again you can inspect the result:
```

```
Error: unexpected symbol in "Again you"
```

Hide

```
```{r}
```

```
Error: attempt to use zero-length variable name
```

Hide

```
nn2$result.matrix
```
```

```
                                    [,1]
error                           1.305459e+00
reached.threshold               9.314948e-03
steps                           2.297000e+03
Intercept.to.1layhid1           1.608301e+00
int.to.1layhid1                -1.857785e+00
uv.to.1layhid1                  1.131992e+00
pc.to.1layhid1                 -1.510292e+00
time_spent.to.1layhid1          1.532339e+00
subject.to.1layhid1            -1.015470e+00
enrollment_reason.to.1layhid1 -3.615166e-02
semester.to.1layhid1           -4.042973e-01
cogproc.to.1layhid1             5.893093e-01
social.to.1layhid1             -3.004527e+00
posemo.to.1layhid1              2.962765e-02
negemo.to.1layhid1             -3.864520e+00
n.to.1layhid1                  -3.560328e+00
Intercept.to.1layhid2           6.894249e-01
int.to.1layhid2                 1.669404e+00
uv.to.1layhid2                 -4.930519e-01
pc.to.1layhid2                  8.893686e-01
time_spent.to.1layhid2          1.543027e+00
subject.to.1layhid2             3.371560e-01
enrollment_reason.to.1layhid2 -6.289845e-01
semester.to.1layhid2            9.526976e-01
cogproc.to.1layhid2            -4.289560e-01
social.to.1layhid2              3.166546e-01
posemo.to.1layhid2             -7.078261e+00
negemo.to.1layhid2             -3.752829e+00
n.to.1layhid2                  -1.388962e+01
Intercept.to.1layhid3          -2.588354e-01
int.to.1layhid3                -8.007996e-01
uv.to.1layhid3                  1.065948e+00
pc.to.1layhid3                 -2.334402e+00
time_spent.to.1layhid3          5.083887e+00
subject.to.1layhid3             1.311230e+00
enrollment_reason.to.1layhid3  2.153428e-01
semester.to.1layhid3            2.475391e+00
cogproc.to.1layhid3            -3.551153e-01
social.to.1layhid3             -2.247629e-01
posemo.to.1layhid3              3.174167e-01
negemo.to.1layhid3             -2.223564e+00
n.to.1layhid3                  -7.294742e+00
Intercept.to.1layhid4           1.238559e-01
int.to.1layhid4                -1.204580e+00
uv.to.1layhid4                 -2.296949e-01
pc.to.1layhid4                 -9.096160e-01
time_spent.to.1layhid4          4.146454e+00
subject.to.1layhid4            -6.080333e-01
enrollment_reason.to.1layhid4  6.350847e-01
semester.to.1layhid4           -1.236434e+00
```

```
cogproc.to.1layhid4            2.962644e-01
social.to.1layhid4            -2.032994e+00
posemo.to.1layhid4            -3.277456e+00
negemo.to.1layhid4            -4.892093e-01
n.to.1layhid4                  6.322623e+00
Intercept.to.1layhid5          1.359490e+00
int.to.1layhid5                1.188555e+00
uv.to.1layhid5                -1.918540e+00
pc.to.1layhid5                -1.137538e+00
time_spent.to.1layhid5         2.964095e+01
subject.to.1layhid5           -7.267960e+00
enrollment_reason.to.1layhid5  1.205480e+00
semester.to.1layhid5           6.172854e+00
cogproc.to.1layhid5           -6.589797e+00
social.to.1layhid5             9.116563e+00
posemo.to.1layhid5            -3.574201e+01
negemo.to.1layhid5             1.040068e+01
n.to.1layhid5                  1.234409e+01
Intercept.to.2layhid1          4.624593e-01
1layhid1.to.2layhid1          -4.214670e+00
1layhid2.to.2layhid1          -3.527544e+00
1layhid3.to.2layhid1           1.077821e+00
1layhid4.to.2layhid1           1.761468e+00
1layhid5.to.2layhid1          -1.404635e+00
Intercept.to.2layhid2          1.684287e-01
1layhid1.to.2layhid2           2.792868e+01
1layhid2.to.2layhid2          -3.709084e+01
1layhid3.to.2layhid2          -1.486047e+01
1layhid4.to.2layhid2          -1.883509e+00
1layhid5.to.2layhid2          -5.997832e+00
Intercept.to.2layhid3         -6.464645e+00
1layhid1.to.2layhid3           4.065486e+01
1layhid2.to.2layhid3          -8.920467e+00
1layhid3.to.2layhid3          -1.479811e+00
1layhid4.to.2layhid3          -9.773548e+00
1layhid5.to.2layhid3           3.812878e-01
Intercept.to.2layhid4          1.681695e+00
1layhid1.to.2layhid4           3.577514e+00
1layhid2.to.2layhid4           1.110199e+01
1layhid3.to.2layhid4          -4.508890e+00
1layhid4.to.2layhid4          -2.982119e+01
1layhid5.to.2layhid4          -1.093998e+01
Intercept.to.2layhid5         -4.844262e-01
1layhid1.to.2layhid5          -1.595438e+01
1layhid2.to.2layhid5          -2.978120e+00
1layhid3.to.2layhid5           4.253929e+00
1layhid4.to.2layhid5           1.110441e+01
1layhid5.to.2layhid5          -7.750785e+01
Intercept.to.final_grade       1.092924e-01
2layhid1.to.final_grade        3.171659e+00
2layhid2.to.final_grade       -8.518346e+00
```

```
2layhid3.to.final_grade       -5.303078e+00
2layhid4.to.final_grade       -4.016515e+00
2layhid5.to.final_grade        3.601584e+00
```

Hide

```
<!-- rnb-source-end -->

<!-- rnb-output-begin eyJkYXRhIjoiRXJyb3I6IGF0dGVtcHQgdG8gdXNlIHplcm8tbGVuZ3RoIHZhcml
hYmxlIG5hbWVcbiJ9 -->
```

Error: attempt to use zero-length variable name

```
<!-- rnb-output-end -->

<!-- rnb-chunk-end -->


<!-- rnb-text-begin -->


Again you can inspect the result:

<!-- rnb-text-end -->


<!-- rnb-chunk-begin -->

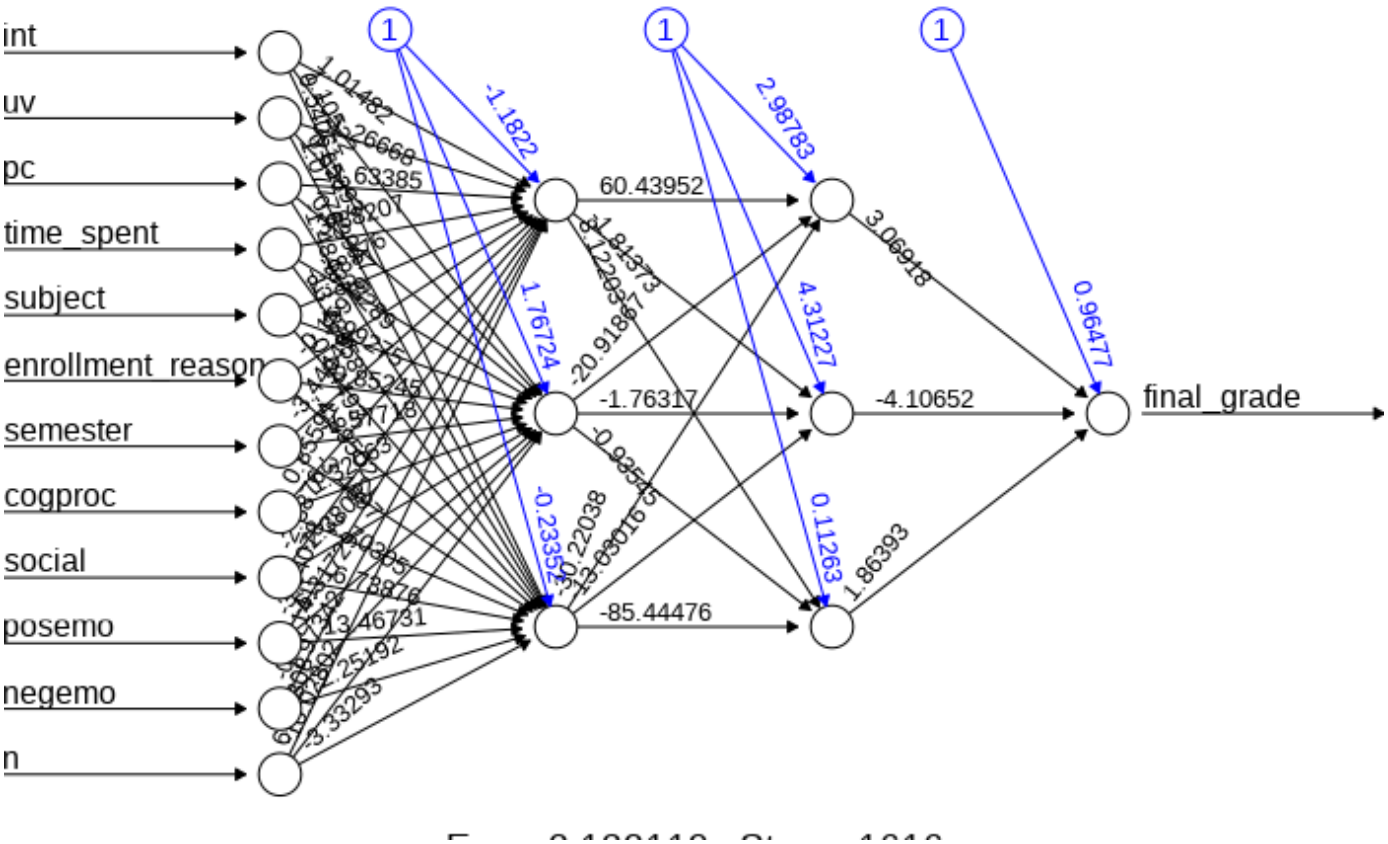

<!-- rnb-source-begin eyJkYXRhIjoiYGBgclxubm4yJHJlc3VsdC5tYXRyaXhcblxuYGBgIn0= -->

```r
nn2$result.matrix
```
```

```
                                    [,1]
error                           2.132119e+00
reached.threshold               9.585912e-03
steps                           1.616000e+03
Intercept.to.1layhid1          -1.182203e+00
int.to.1layhid1                 1.014822e+00
uv.to.1layhid1                 -1.266685e+00
pc.to.1layhid1                 -1.633850e+00
time_spent.to.1layhid1          3.882071e+00
subject.to.1layhid1             2.188760e+00
enrollment_reason.to.1layhid1 -1.591855e-01
semester.to.1layhid1           -3.444459e+00
cogproc.to.1layhid1             8.559105e-01
social.to.1layhid1             -2.387604e+00
posemo.to.1layhid1             -3.940781e+00
negemo.to.1layhid1             -9.714086e-01
n.to.1layhid1                   6.568078e+00
Intercept.to.1layhid2           1.767242e+00
int.to.1layhid2                 1.105222e+00
uv.to.1layhid2                  6.539336e-01
pc.to.1layhid2                  8.394727e-01
time_spent.to.1layhid2         -9.382891e+00
subject.to.1layhid2             9.927468e-01
enrollment_reason.to.1layhid2 -8.524527e-01
semester.to.1layhid2           -1.577184e+00
cogproc.to.1layhid2             1.320634e+00
social.to.1layhid2             -2.380817e+00
posemo.to.1layhid2             -6.317290e+00
negemo.to.1layhid2             -1.973419e+00
n.to.1layhid2                  -1.002802e+01
Intercept.to.1layhid3          -2.335189e-01
int.to.1layhid3                 5.406130e-01
uv.to.1layhid3                  1.077763e+00
pc.to.1layhid3                 -1.484362e+00
time_spent.to.1layhid3         -1.323958e+01
subject.to.1layhid3            -4.493006e-02
enrollment_reason.to.1layhid3 -4.895017e+00
semester.to.1layhid3           -3.489698e+00
cogproc.to.1layhid3            -2.703051e+00
social.to.1layhid3             -6.738765e+00
posemo.to.1layhid3              1.346731e+01
negemo.to.1layhid3             -2.251924e+00
n.to.1layhid3                  -3.332928e+00
Intercept.to.2layhid1           2.987833e+00
1layhid1.to.2layhid1            6.043952e+01
1layhid2.to.2layhid1           -2.091867e+01
1layhid3.to.2layhid1           -3.022038e+01
Intercept.to.2layhid2           4.312269e+00
1layhid1.to.2layhid2           -1.813729e+00
1layhid2.to.2layhid2           -1.763168e+00
1layhid3.to.2layhid2           -1.303016e+01
```

```
Intercept.to.2layhid3          1.126252e-01
1layhid1.to.2layhid3           8.122027e+00
1layhid2.to.2layhid3          -9.354471e-01
1layhid3.to.2layhid3          -8.544476e+01
Intercept.to.final_grade       9.647669e-01
2layhid1.to.final_grade        3.069178e+00
2layhid2.to.final_grade       -4.106521e+00
2layhid3.to.final_grade        1.863926e+00
```

and the plot shows what our parameters changed:



and finally what level of accuracy did we get with the testing data:

actual <dbl>	prediction <dbl>
0.93871751	0.834652045
0.65658904	0.864349854
0.73643780	0.802709908
0.80630822	0.879923230
0.76851380	0.828720976
0.94763665	0.904155185
0.83142731	0.897230198

actual <dbl>	prediction <dbl>
0.98638316	0.875457634
0.91376004	0.873540533
0.87175506	0.829638387

1-10 of 92 rows Previous **1** 2 3 4 5 6 ... 10 Next

Hide

```
accuracy(results)
```

```
[1] 0.9473571
```

In practice you want to be more systematic about this, and use cross validation in the training data when exploring parameter values - giving the training data a chance to perform as both training and testing during the parameter selection phase. Once you have selected your final model is when you would return to the original testing data.

Unfortunately the train function we used for random forest models to implement cross validation for model selection does not work with neural networks.

You can find instructions for doing it manually here: https://www.r-bloggers.com/2015/09/fitting-a-neural-network-in-r-neuralnet-package/ (https://www.r-bloggers.com/2015/09/fitting-a-neural-network-in-r-neuralnet-package/)