Garg_Bheeni_Stat6620_Homework6

Bheeni Garg

May 31, 2016

Question 1: Neural Networks ————————-

Example: Modeling the Strength of Concrete —-

Step 1 : Collecting Data

The concrete dataset contains 1,030 examples of concrete, with eight features describing the components used in the mixture. These features are thought to be related to the nal compressive strength, and they include the amount (in kilograms per cubic meter) of cement, slag, ash, water, superplasticizer, coarse aggregate, and ne aggregate used in the product, in addition to the aging time (measured in days).

Step 2: Exploring and preparing the data —-

```
# read in data and examine structure
concrete <- read.csv("concrete.csv")
str(concrete)
```

```
## 'data.frame':    1030 obs. of  9 variables:
##  $ cement      : num  141 169 250 266 155 ...
##  $ slag        : num  212 42.2 0 114 183.4 ...
##  $ ash         : num  0 124.3 95.7 0 0 ...
##  $ water       : num  204 158 187 228 193 ...
##  $ superplastic: num  0 10.8 5.5 0 9.1 0 0 6.4 0 9 ...
##  $ coarseagg   : num  972 1081 957 932 1047 ...
##  $ fineagg     : num  748 796 861 670 697 ...
##  $ age         : int  28 14 28 28 28 90 7 56 28 28 ...
##  $ strength    : num  29.9 23.5 29.2 45.9 18.3 ...
```

```
# custom normalization function
normalize <- function(x) {
  return((x - min(x)) / (max(x) - min(x)))
}

# apply normalization to entire data frame
concrete_norm <- as.data.frame(lapply(concrete, normalize))

# confirm that the range is now between zero and one
summary(concrete_norm$strength)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.2664  0.4001  0.4172  0.5457  1.0000
```

```
# compared to the original minimum and maximum
summary(concrete$strength)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.33   23.71   34.44   35.82   46.14   82.60
```

```
# create training and test data
concrete_train <- concrete_norm[1:773, ]
concrete_test <- concrete_norm[774:1030, ]
```

Step 3: Training a model on the data —-

```
# train the neuralnet model
library(neuralnet)
```

```
## Loading required package: grid
```

```
## Loading required package: MASS
```

```
# simple ANN with only a single hidden neuron
set.seed(12345) # to guarantee repeatable results
concrete_model <- neuralnet(formula = strength ~ cement + slag +
                              ash + water + superplastic +
                              coarseagg + fineagg + age,
                              data = concrete_train)

# visualize the network topology
plot(concrete_model)

# Reference: http://www.r-bloggers.com/neuralnettools-1-0-0-now-on-cran/
# alternative plot
library(NeuralNetTools)

# plotnet
par(mar = numeric(4), family = 'serif')
plotnet(concrete_model, alpha = 0.6)
```

Step 4: Evaluating model performance —-

```
# obtain model results
model_results <- compute(concrete_model, concrete_test[1:8])
# obtain predicted strength values
predicted_strength <- model_results$net.result
# examine the correlation between predicted and actual values
cor(predicted_strength, concrete_test$strength)
```

```
##              [,1]
## [1,] 0.8064655576
```

Step 5: Improving model performance —-

```
# a more complex neural network topology with 5 hidden neurons
set.seed(12345) # to guarantee repeatable results
concrete_model2 <- neuralnet(strength ~ cement + slag +
                               ash + water + superplastic +
                               coarseagg + fineagg + age,
                               data = concrete_train, hidden = 5)

# plot the network
plot(concrete_model2)

# plotnet
par(mar = numeric(4), family = 'serif')
plotnet(concrete_model2, alpha = 0.6)

# evaluate the results as we did before
model_results2 <- compute(concrete_model2, concrete_test[1:8])
predicted_strength2 <- model_results2$net.result
cor(predicted_strength2, concrete_test$strength)
```

```
##              [,1]
## [1,] 0.9244533426
```

Question 2: Logistic Regression—-

The Stock Market Data

```
library(ISLR)
names(Smarket)
```

```
## [1] "Year"      "Lag1"      "Lag2"      "Lag3"      "Lag4"      "Lag5"
## [7] "Volume"    "Today"     "Direction"
```

```
dim(Smarket)
```

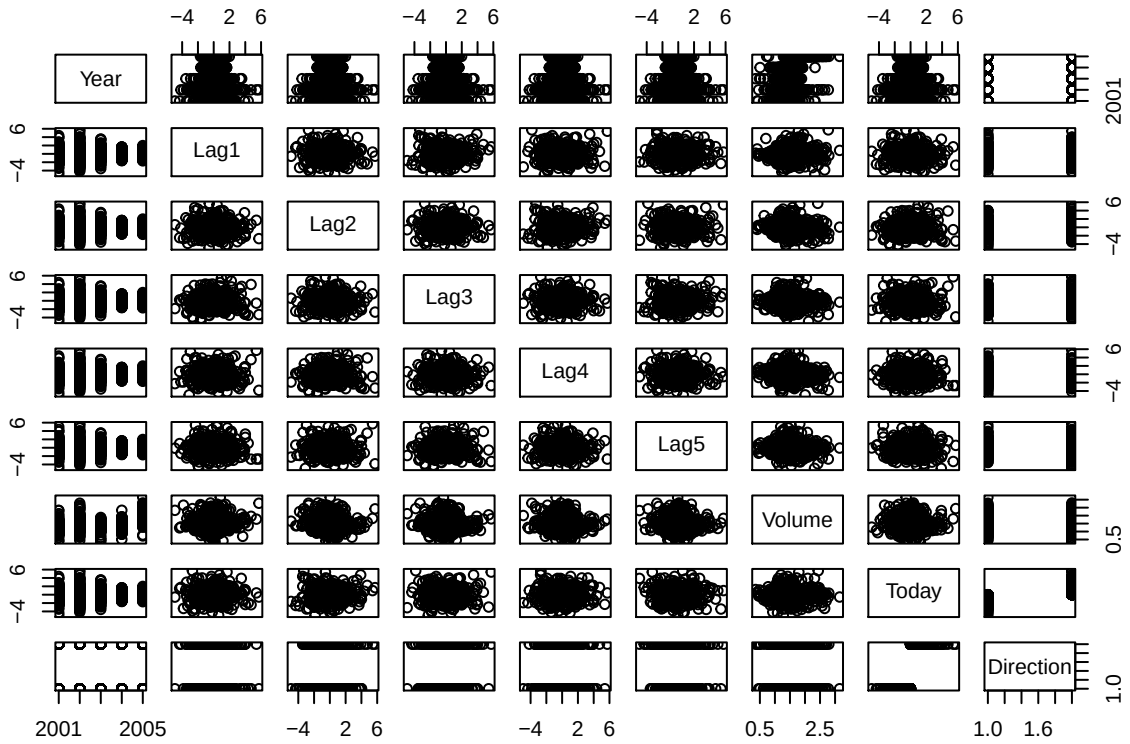
```
## [1] 1250    9
```

```
summary(Smarket)
```

```
##       Year               Lag1                Lag2
##  Min.   :2001.000   Min.   :-4.9220000   Min.   :-4.9220000
##  1st Qu.:2002.000   1st Qu.:-0.6395000   1st Qu.:-0.6395000
##  Median :2003.000   Median : 0.0390000   Median : 0.0390000
##  Mean   :2003.016   Mean   : 0.0038344   Mean   : 0.0039192
##  3rd Qu.:2004.000   3rd Qu.: 0.5967500   3rd Qu.: 0.5967500
##  Max.   :2005.000   Max.   : 5.7330000   Max.   : 5.7330000
##       Lag3                Lag4                Lag5
```

```
##  Min.   :-4.922000   Min.   :-4.922000   Min.   :-4.9220000
##  1st Qu.:-0.640000   1st Qu.:-0.640000   1st Qu.:-0.6400000
##  Median : 0.038500   Median : 0.038500   Median : 0.0385000
##  Mean   : 0.001716   Mean   : 0.001636   Mean   : 0.0056096
##  3rd Qu.: 0.596750   3rd Qu.: 0.596750   3rd Qu.: 0.5970000
##  Max.   : 5.733000   Max.   : 5.733000   Max.   : 5.7330000
##      Volume             Today            Direction
##  Min.   :0.356070   Min.   :-4.9220000   Down:602
##  1st Qu.:1.257400   1st Qu.:-0.6395000   Up  :648
##  Median :1.422950   Median : 0.0385000
##  Mean   :1.478305   Mean   : 0.0031384
##  3rd Qu.:1.641675   3rd Qu.: 0.5967500
##  Max.   :3.152470   Max.   : 5.7330000
```

```
pairs(Smarket)
```

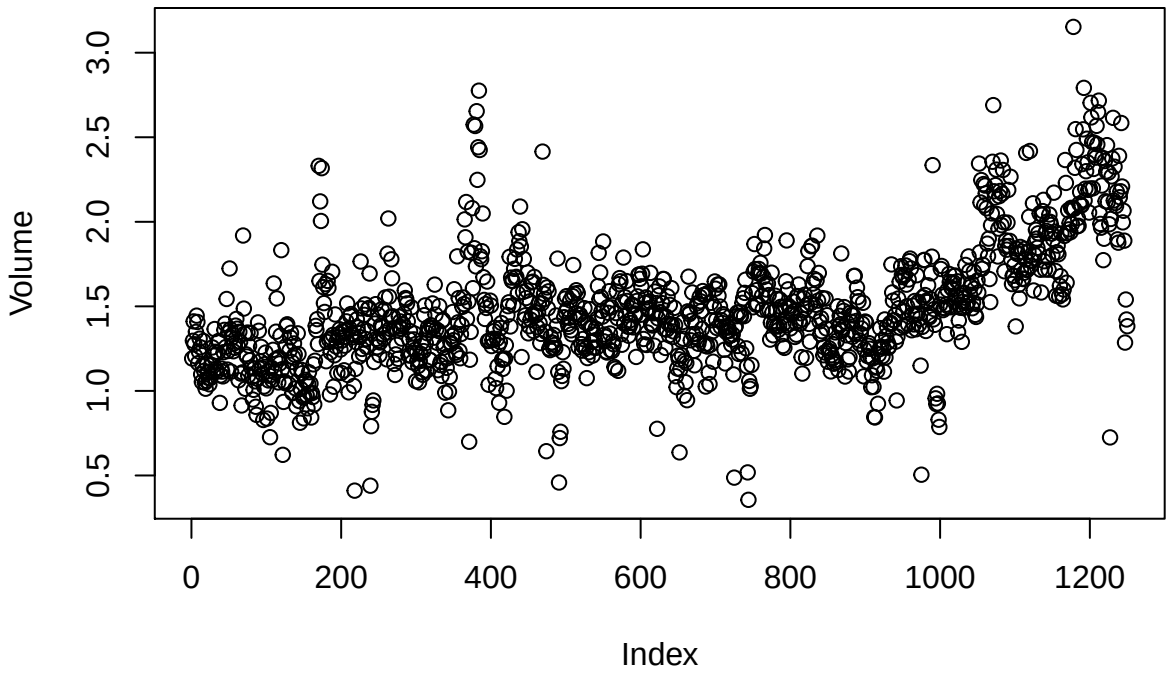



```
cor(Smarket[,-9])
```

```
##                 Year           Lag1           Lag2           Lag3
## Year    1.00000000000  0.029699648934  0.030596421947  0.033194581131
## Lag1    0.02969964893  1.000000000000 -0.026294328150 -0.010803401705
## Lag2    0.03059642195 -0.026294328150  1.000000000000 -0.025896669813
## Lag3    0.03319458113 -0.010803401705 -0.025896669813  1.000000000000
## Lag4    0.03568871750 -0.002985910700 -0.010853533001 -0.024051036377
## Lag5    0.02978799469 -0.005674606239 -0.003557949478 -0.018808337582
## Volume  0.53900646614  0.040909908067 -0.043383214642 -0.041823686275
## Today   0.03009522893 -0.026155045432 -0.010250033403 -0.002447647117
##                 Lag4           Lag5         Volume           Today
## Year    0.035688717504  0.029787994692  0.53900646614  0.030095228929
```

```
## Lag1   -0.002985910700 -0.005674606239  0.04090990807 -0.026155045432
## Lag2   -0.010853533001 -0.003557949478 -0.04338321464 -0.010250033403
## Lag3   -0.024051036377 -0.018808337582 -0.04182368628 -0.002447647117
## Lag4    1.000000000000 -0.027083640769 -0.04841424624 -0.006899526938
## Lag5   -0.027083640769  1.000000000000 -0.02200231487 -0.034860082787
## Volume -0.048414246240 -0.022002314872  1.00000000000  0.014591823259
## Today  -0.006899526938 -0.034860082787  0.01459182326  1.000000000000
```

```
attach(Smarket)
plot(Volume)
```



Logistic Regression

```
glm.fit=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Smarket,family=binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial, data = Smarket)
##
## Deviance Residuals:
##       Min         1Q     Median         3Q        Max
## -1.446343  -1.203130   1.065292   1.145081   1.325832
##
## Coefficients:
##                 Estimate   Std. Error  z value Pr(>|z|)
## (Intercept) -0.126000257  0.240735738 -0.52340  0.60070
## Lag1        -0.073073746  0.050167387 -1.45660  0.14523
```

```
## Lag2        -0.042301344  0.050086051 -0.84457  0.39835
## Lag3         0.011085108  0.049938545  0.22197  0.82433
## Lag4         0.009358938  0.049974131  0.18728  0.85144
## Lag5         0.010313068  0.049511460  0.20830  0.83500
## Volume       0.135440659  0.158359698  0.85527  0.39240
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1731.1748  on 1249  degrees of freedom
## Residual deviance: 1727.5841  on 1243  degrees of freedom
## AIC: 1741.5841
##
## Number of Fisher Scoring iterations: 3
```

```
coef(glm.fit)
```

```
##    (Intercept)           Lag1           Lag2           Lag3           Lag4
## -0.12600025656 -0.07307374589 -0.04230134401  0.01108510838  0.00935893837
##           Lag5         Volume
##  0.01031306848  0.13544065886
```

```
summary(glm.fit)$coef
```

```
##                   Estimate    Std. Error       z value     Pr(>|z|)
## (Intercept) -0.12600025656 0.24073573811 -0.5233965574 0.6006983194
## Lag1        -0.07307374589 0.05016738680 -1.4565986102 0.1452272116
## Lag2        -0.04230134401 0.05008605132 -0.8445733470 0.3983490954
## Lag3         0.01108510838 0.04993854467  0.2219749985 0.8243333461
## Lag4         0.00935893837 0.04997413139  0.1872756586 0.8514445069
## Lag5         0.01031306848 0.04951145984  0.2082965945 0.8349973905
## Volume       0.13544065886 0.15835969787  0.8552722737 0.3924004332
```

```
summary(glm.fit)$coef[,4]
```

```
##  (Intercept)         Lag1         Lag2         Lag3         Lag4
## 0.6006983194 0.1452272116 0.3983490954 0.8243333461 0.8514445069
##         Lag5       Volume
## 0.8349973905 0.3924004332
```

```
glm.probs=predict(glm.fit,type="response")
glm.probs[1:10]
```

```
##           1           2           3           4           5
## 0.5070841334 0.4814678785 0.4811388352 0.5152223558 0.5107811627
##           6           7           8           9          10
## 0.5069564605 0.4926508742 0.5092291582 0.5176135262 0.4888377798
```

```
contrasts(Direction)
```

```
##      Up
## Down  0
## Up    1
```

```
glm.pred=rep("Down",1250)
glm.pred[glm.probs>.5]="Up"
table(glm.pred,Direction)
```

```
##         Direction
## glm.pred Down  Up
##     Down  145 141
##     Up    457 507
```

```
(507+145)/1250
```

```
## [1] 0.5216
```

```
mean(glm.pred==Direction)
```

```
## [1] 0.5216
```

```
train=(Year<2005)
Smarket.2005=Smarket[!train,]
dim(Smarket.2005)
```

```
## [1] 252   9
```

```
Direction.2005=Direction[!train]
glm.fit=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Smarket,family=binomial,subset=train)
glm.probs=predict(glm.fit,Smarket.2005,type="response")
glm.pred=rep("Down",252)
glm.pred[glm.probs>.5]="Up"
table(glm.pred,Direction.2005)
```

```
##         Direction.2005
## glm.pred Down Up
##     Down   77 97
##     Up     34 44
```

```
mean(glm.pred==Direction.2005)
```

```
## [1] 0.4801587302
```

```
mean(glm.pred!=Direction.2005)
```

```
## [1] 0.5198412698
```

```
glm.fit=glm(Direction~Lag1+Lag2,data=Smarket,family=binomial,subset=train)
glm.probs=predict(glm.fit,Smarket.2005,type="response")
glm.pred=rep("Down",252)
glm.pred[glm.probs>.5]="Up"
table(glm.pred,Direction.2005)
```

```
##         Direction.2005
## glm.pred Down  Up
##     Down   35  35
##     Up     76 106
```

```
mean(glm.pred==Direction.2005)
```

```
## [1] 0.5595238095
```

```
106/(106+76)
```

```
## [1] 0.5824175824
```

```
predict(glm.fit,newdata=data.frame(Lag1=c(1.2,1.5),Lag2=c(1.1,-0.8)),type="response")
```

```
##           1           2
## 0.4791462392 0.4960938730
```
