Data Mining: Assignment Two

Aaron Gonzales

October 25, 2014

1 Neural Network

Reading the data, optimizing for speed:

```
tab5rows <- read.table("~/Dropbox/cs/datamining/data-mining-is-fun/assignments/two/data.txt", header = FALSE, nr
classes <- sapply(tab5rows, class)
data <- read.table("~/Dropbox/cs/datamining/data-mining-is-fun/assignments/two/data.txt", header = FALSE, colCla
```

(a)

Using the nnet package and a small function to tabulate our predictions.

```
library(nnet)
library(caret)

## Loading required package:  lattice

library(pROC)

## Type 'citation("pROC")' for a citation.
##
## Attaching package:  'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var

nrow(data)

## [1] 10000

#Define class 'labels' for our set.
targets <- class.ind(c(rep('0', 5000), rep('1',5000)))
labels <- c(rep(0,5000), rep(1,5000))
data$labels <- labels
data$labels <- as.factor(data$labels)

# small function to show predicted results
test.cl <- function(true, pred) {
    actual <- max.col(true) - 1
    predicted <- max.col(pred) - 1
    table(actual, predicted)

}
```

```
# caret has many functions to help chop up data, here we partition the data
# into a test and training set
library(doMC)

## Loading required package:  foreach
## Loading required package:  iterators
## Loading required package:  parallel

# sets multiple cores to help run the svm
```

```
registerDoMC(2)
data_small <- subset(data, select=c(V1, V2, V3, V4, labels))
inTrain <- createDataPartition(y = data_small$labels, p=.75, list=FALSE)
training <- data_small[inTrain,]
testing <- data_small[-inTrain,]

# here is a control model that specifies to use repeated 10-fold cross validation
ctrl <- trainControl(method = 'repeatedcv',
                     number = 10
                     )

# we train the model on the training data and with the labeling data as a guide
fitsmall <- train( training, training$labels,
              method = "nnet", # uses the 'nnet' package as a backend
              algorithm = 'backprop',
              learningrate = 0.1,
              trControl = ctrl,
              linout = FALSE,
              MaxNWts = 15000)
## many pages of printing supressed
```

```
#shows the confusion matrix
confusionMatrix(fitsmall)

## Cross-Validated (10 fold, repeated 1 times) Confusion Matrix
##
## (entries are percentages of table totals)
##
##           Reference
## Prediction  0  1
##          0 50  0
##          1  0 50

#prediction
smallpre <- predict(fitsmall, newdata = testing)
summary(smallpre)

##    0    1
## 1250 1250

a <- test.cl(testing$labels, predict(fitsmall, testing))
print(a)

##       predicted
## actual    0
##      0 2500
```

```
# caret has many functions to help chop up data, here we partition the data
# into a test and training set
inTrain <- createDataPartition(y = data$labels, p=.75, list=FALSE)
training <- data[inTrain,]
testing <- data[-inTrain,]
```

```
# here is a control model that specifies to use repeated 10-fold cross validation
ctrl <- trainControl(method = 'cv',
                     number = 10
                     )

# we train the model on the training data and with the labeling data as a guide
fitnn <- train( training, training$labels,
              method = "nnet", # uses the 'nnet' package as a backend
              algorithm = 'backprop',
              learningrate = 0.2,
              trControl = ctrl,
              linout = FALSE,
              MaxNWts = 15000,
              maxit = 100,
              hidden = 10)
## many pages of printing supressed
```

```
#shows the confusion matrix
confusionMatrix(fitnn)

## Cross-Validated (10 fold) Confusion Matrix
##
## (entries are percentages of table totals)
##
##            Reference
## Prediction    0    1
##           0 45.2  2.6
##           1  4.8 47.4

#prediction
pre <- predict(fitnn, newdata = testing)
summary(pre)

##    0    1
## 1226 1274

a <- test.cl(testing$labels, predict(fitnn, testing))
print(a)

##       predicted
## actual    0
##      0 2500
```

Neural network...

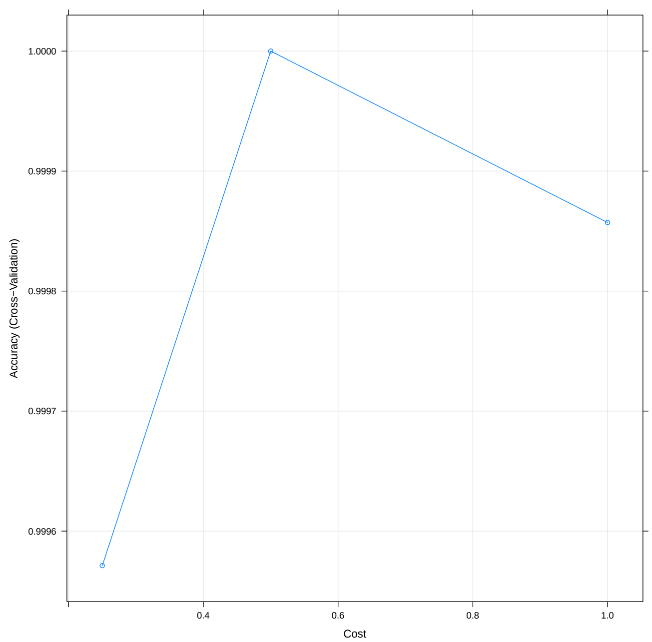
We use the caret (classification and regression tool) package with doMC(Parallelization for R) to train an svm (libsvm package) with a radial basis function.

```
library(doMC)
# sets multiple cores to help run the svm
registerDoMC(2)
# subsetting into train and test sets.
index <- 1:nrow(data)
testindex <- sample(index, trunc(length(index)*30/100))
testset <- data[testindex,]
```

```
trainset <- data[-testindex,]

# training the model with a radial basis function.
# uses the lables ~ . to say 'the class labels are defined in the varible
# 'labels'. 10-fold cross validation is performed in the model and reported later and
# doesn't need to be done by hand.
system.time(
  fitsvm <- train(labels ~ .,
                  data = trainset,
                  method="svmRadial",
                  trControl=trainControl(method='cv', number = 10)
                  )
)

##    user  system elapsed
## 789.785   7.378 915.254
```

```
print(fitsvm)

## Support Vector Machines with Radial Basis Function Kernel
##
## 7000 samples
## 1000 predictors
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
##
## Summary of sample sizes: 6300, 6300, 6301, 6300, 6300, 6300, ...
##
## Resampling results across tuning parameters:
##
##   C    Accuracy  Kappa  Accuracy SD  Kappa SD
##   0.2  1         1      1e-03        2e-03
##   0.5  1         1      0e+00        0e+00
##   1.0  1         1      5e-04        9e-04
##
## Tuning parameter 'sigma' was held constant at a value of 0.0005525
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were sigma = 0.0005525 and C = 0.5.

# prediction.
prediction <- predict(fitsvm, testset[,-1001])
tab <- table(pred = prediction, true = testset$labels)
plot(fitsvm)
confusionMatrix(fitsvm)

## Cross-Validated (10 fold) Confusion Matrix
##
## (entries are percentages of table totals)
##
##           Reference
## Prediction    0    1
##          0 49.6  0.0
##          1  0.0 50.4
```

```
print(tab)

##      true
## pred    0    1
##    0 1529    0
##    1    0 1471

# model accuracy
print(1-(tab[1,2]+tab[2,1]) / (tab[1,1]+tab[2,2]))

## [1] 1
```



2 Scrambled data

Here we assign the classes according to the assignment rubric, divided in chunks of 500.

```
data_copy <- data

labels <- c(
  rep('0',500),
        rep('1',500),
        rep('0',500),
        rep('1',500),
        rep('0',500),
        rep('1',500),
        rep('0',500),
        rep('1',500),
        rep('0',500),
        rep('1',500),
        rep('0',500),
        rep('1',500),
        rep('0',500),
```

```
        rep('1',500),
        rep('0',500),
        rep('1',500),
        rep('0',500),
        rep('1',500),
        rep('0',500),
        rep('1',500)
)

data_copy$labels <- as.factor(labels)
is.factor(data_copy$labels)

## [1] TRUE
```

(a) Neural Network, again

```
inTrain <- createDataPartition(y = data_copy$labels, p=.71, list=FALSE)
training <- data_copy[inTrain,]
testing <- data_copy[-inTrain,]

ctrl <- trainControl(method = 'repeatedcv',
                     number = 10
                     )

is.factor(training$labels)
fitnn2 <- train( training, training$labels,
              method = "nnet",
              algorithm = 'backprop',
              learningrate = 0.1,
              trControl = ctrl,
              linout = FALSE,
              ## this was the important thing to pass to nnet
              MaxNWts = 15000)

## many pages of printing output is supressed here
```

```
confusionMatrix(fitnn2)

## Cross-Validated (10 fold, repeated 1 times) Confusion Matrix
##
## (entries are percentages of table totals)
##
##            Reference
## Prediction    0    1
##          0 46.2  2.8
##          1  3.8 47.2

#prediction
pre <- predict(fitnn2, newdata = testing)
summary(pre)

##    0    1
## 1445 1455
```

```
a <- test.cl(testing$labels, predict(fitnn2, testing))
print(a)

##       predicted
## actual    0
##      0 2900
```

(b) SVM, again

```
# subsetting into train and test sets.
index <- 1:nrow(data_copy)
testindex <- sample(index, trunc(length(index)*30/100))
testset <- data_copy[testindex,]
trainset <- data_copy[-testindex,]

# training the model with a radial basis function.
# uses the lables ~ . to say 'the class labels are defined in the varible
# 'labels'. 10-fold cross validation is performed in the model and reported later and
# doesn't need to be done by hand.
system.time(
  model <- train(labels ~ .,
                 data = trainset,
                 method="svmRadial",
                 trControl=trainControl(method='cv', number = 10)
                 )
)

## Loading required package:  kernlab

##    user  system elapsed
## 4286.61   27.85 4347.85
```

```
print(model)

## Support Vector Machines with Radial Basis Function Kernel
##
## 7000 samples
## 1000 predictors
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
##
## Summary of sample sizes: 6299, 6300, 6300, 6300, 6300, 6300, ...
##
## Resampling results across tuning parameters:
##
##   C    Accuracy  Kappa  Accuracy SD  Kappa SD
##   0.2  0.5       -0.01  0.007        0.01
##   0.5  0.5       -0.02  0.017        0.03
##   1.0  0.5       -0.02  0.015        0.03
##
```

```
## Tuning parameter 'sigma' was held constant at a value of 0.0005511
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were sigma = 0.0005511 and C = 0.25.

# prediction.
prediction <- predict(model, testset[,-1001])
tab <- table(pred = prediction, true = testset$labels)
plot(model)
confusionMatrix(model)

## Cross-Validated (10 fold) Confusion Matrix
##
## (entries are percentages of table totals)
##
##           Reference
## Prediction    0    1
##          0  4.1  4.9
##          1 45.3 45.6

print(tab)

##     true
## pred    0    1
##    0  153  154
##    1 1386 1307

# model accuracy
print(1-(tab[1,2]+tab[2,1]) / (tab[1,1]+tab[2,2]))

## [1] -0.05479
```