HW 10

Ian Douglas

5/13/2019

# read in the data

df10 <- read.table("http://users.stat.ufl.edu/~rrandles/sta4210/Rclassnotes/data/textdatasets/KutnerData/Appendix%20C%20Data%20Sets/APPENC07.txt")  
vnames <- c("sales.price", "sq.ft", "num.bed", "num.bath",   
 "ac", "g.size", "pool", "year.build", "quality",   
 "style", "l.size", "adj.hway")  
#Removing ID number  
df10 <- df10[,-1]  
names(df10) <- vnames  
#change unites of price variable in terms of dollars  
df10$sales.price <- (df10$sales.price/1000000)  
#rename y variable  
names(df10) <- c("y",names(df10)[-1])  
#Creating formula for RMSE  
rmse.nn <- function(model,test.on=test.df10) {  
 sqrt(mean((predict(model,newdata=test.on)-test.on[,"y"])^2))  
}  
#scale entire dataset  
maxs10 <- apply(df10, 2, max)  
mins10 <- apply(df10, 2, min)  
df10.scl <- as.data.frame(scale(df10, center = mins10,   
 scale = maxs10 - mins10))

1. **Select a random sample of 300 observations to use as a training dataset.**

set.seed(6)  
ii <- sample(1:nrow(df10.scl), size = 300, replace = FALSE)  
train.df10 <- df10.scl[ii,]  
test.df10 <- df10.scl[-ii,]

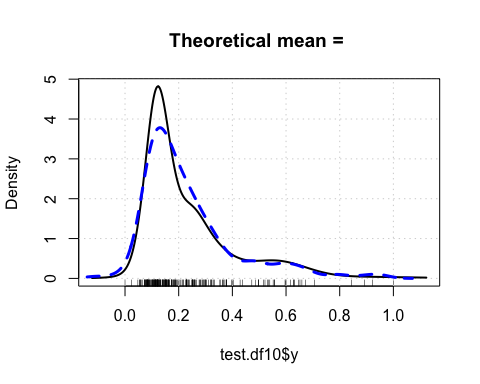
1. **Develop a neural network model for predicting sales price. Try your best to find** a good number of hidden nodes and other tuning parameters.

rmse.nn <- function(model, test.on=test.df10) {  
 sqrt(mean((predict(model, newdata=test.on)- test.on[,1])^2))  
}  
nn.results <- c()  
thresh <- seq(.009, .011,by = .001)  
for (i in 1:15) {  
 for (j in 1:length(thresh)){  
 model.nn <- neuralnet(  
 y ~ sq.ft + num.bed + num.bath +ac + g.size + pool  
 + year.build + quality + style +l.size + adj.hway,  
 train.df10, hidden = i, threshold = thresh[j]  
 )  
 rmse.tmp <- rmse.nn(model = model.nn)  
 nn.results <- c(nn.results, rmse.tmp)  
 }  
}  
best <- data.frame("layers"=rep(1:15,length.out=length(nn.results)),  
 "threshold"=rep(thresh,length.out=length(nn.results)),  
 "rmse" = nn.results)[which.min(nn.results),]  
best #3 layers at threshold .011; rmse = .075

## layers threshold rmse  
## 18 3 0.011 0.0750622

1. Assess your model’s ability to predict and discuss its usefulness as a tool for predicting sales prices. (Here you need to use the test dataset.)

best.net <- neuralnet(  
 y ~ sq.ft + num.bed + num.bath +ac + g.size + pool  
 + year.build + quality + style +l.size + adj.hway,  
 train.df10, hidden = 3, threshold = .011  
 )  
  
car::densityPlot(test.df10$y, col = "black",  
 main = "Theoretical mean = ")  
lines(density(predict(best.net, newdata=test.df10)),col = "blue",  
 lty= 2, lwd = 3)



It looks like the neural net actuall underpredicts sales around the mean. Though, it is possible that this density plot has a poor bandwith.

#to supplement, here is the mean and SD of the predicted and actual:  
data.frame(test.df10$y, predict(best.net, newdata=test.df10)) %>% apply(2, function(x) rbind(mean(x),sd(x)))

## test.df10.y predict.best.net..newdata...test.df10.  
## Mean 0.2371188 0.2335563  
## SD 0.1797588 0.1760545

The mean of the predicted distribution is fairly accurate, as is the variance.

1. Compare your neural network to a regression model with your choice of best subset selection method. Which model is easier to interpret?

Xy.df10 <- data.frame(df10[-grep("y",names(df10))],df10$y)  
best.reg <- bestglm(Xy.df10)  
summary(best.reg$BestModel)

##   
## Call:  
## lm(formula = y ~ ., data = data.frame(Xy[, c(bestset[-1], FALSE),   
## drop = FALSE], y = y))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.19306 -0.04066 -0.00604 0.02793 0.35795   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.131e-01 1.462e-02 -7.736 5.43e-14 \*\*\*  
## sq.ft 1.281e-04 7.397e-06 17.312 < 2e-16 \*\*\*  
## num.bed -1.479e-02 4.013e-03 -3.687 0.000251 \*\*\*  
## num.bath 1.962e-02 4.899e-03 4.004 7.14e-05 \*\*\*  
## g.size 3.642e-02 5.917e-03 6.156 1.50e-09 \*\*\*  
## l.size 1.008e-06 2.805e-07 3.594 0.000357 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.07354 on 516 degrees of freedom  
## Multiple R-squared: 0.7185, Adjusted R-squared: 0.7157   
## F-statistic: 263.4 on 5 and 516 DF, p-value: < 2.2e-16

(best.reg.lm <- lm(y ~ sq.ft+num.bed+num.bath+g.size+l.size, data = df10))

##   
## Call:  
## lm(formula = y ~ sq.ft + num.bed + num.bath + g.size + l.size,   
## data = df10)  
##   
## Coefficients:  
## (Intercept) sq.ft num.bed num.bath g.size   
## -1.131e-01 1.281e-04 -1.479e-02 1.962e-02 3.642e-02   
## l.size   
## 1.008e-06

The neural net is nearly uninterpretable. In fact, it has zero interpretability relative to the linear model. But it does a better job at predicting.