**Homework Assignment 9 BA288 Murphy**

**Neural Network Models**

Due: Monday, March 20 at 11:59pm.

Worth: 20 points. Two points per part.

***Purpose***

To practice using neural networks to classify spam email. To compare neural networks to other forms of classification models.

***Data Sets***

Spam email is a major challenge for all organizations accounting for 14.5 billion messages per day globally (<https://www.spamlaws.com/spam-stats.html>) or 45% of all emails. This assignment looks at the prediction of spam email, which is generally done automatically, by spam filters.

The data set used in this assignment has 4601 observations. There are 58 variables, with the variable “spam” indicating “1” if the email is spam and “0” if not.

The data, known as the “Spambase Data Set” can be found at the UCI machine learning data repository: <http://archive.ics.uci.edu/ml/datasets/Spambase> The Excel file has information about the variables in the data set.

***Submission***

Submit by upload to Canvas the completed homework assignment. The submission should be

a “.R” file. Later other file submission types may be permitted.

Name the assignment submission “hw9\_lastname\_firstname.R”. The homework should be well-formatted with spacing, well-commented, and easy to find the answers (well-labeled). Use spacing and comments and short lines! No sideways scroll!

**Homework Questions**

The first three questions of this assignment work with a quantitative response variable. The last seven work with a qualitative response.

1. How is spam email normally detected? What are key variables that might indicate that an email is spam? Provide a few brief sentences of insight.

Hint: Open ended question. Research this question in a manner of your choosing.

1. Set a seed. Create training and test data sets. Create the training data set with 2000 observations equally balanced between spam and non-spam observations. Create a test data set with 800 observations again equally balanced.
2. Build a logistic regression model using all the variables in the training data set. Comment on the quality of the model with respect to classification on the training data. Make predictions on the test data, how does the model fitted on the training data perform with respect to classification on this data set?
3. Copy the data to a new data frame for this problem. Normalize the independent variables. Run k-nearest neighbor on the spam data for k = 1 to 99. What value of k gives the lowest test set error?

Hint: The “scale()” function normalizes the data.

Hint 2: Use the loop from the answers to Homework Assignment 7.

1. Create a pruned classification tree on the spam training data. What tree provides the lowest test data set error? What is the error level?

Hint: Copy the original training and test data to a new data frame and convert the response variable in both data frames, that is, the “spam” variable to a factor.

Hint 2: “cv.tree” is a good tool to pick the number of terminal nodes

1. Create a support vector machine (SVM) classifier for the training data using different kernals: linear, radial, polynomial. What is the SVM that provides the best fit on the test data? What is the error level?

Hint: The classifiers known as Support Vector Machines (SVM) are discussed in Chapter 9 of the text.

Hint 2: The “e1071” package is needed for this step.

Hint 3: See the code at the end of the assignment to run a SVM for the linear kernel. The “y” or dependent variable, spam, should be set up as a factor.

Hint 4: Replace the word “linear” with either “radial” or “polynomial” in the “tune” command to build SVMs with the radial and polynomial kernals, respectively.

Hint 5: These routines can take a bit to complete.

1. Build a neural network for classifying each observation as spam. Use a single hidden layer. Try different values for the number of nodes in the hidden layer, 10, 20 and 30. How does the neural network perform with respect to predicting spam?

Hint: The package “neuralnet” is needed for this question and Q8.

Hint 2: The training and test data here must have a numerical y variable. Thus, the original data set will work for this problem.

Hint 3: To set up the formula to run the neural network requires working with strings. See script below for an example.

Hint 4: The script to run the neural net with 1 hidden layer and 10 neurons is shown at the end of the assignment.

1. Try at least one neural network with two hidden layers. How do these models perform on predicting spam on the test data? Report classification errors.

Hint: Use the argument:

hidden = c(20, 10)

inside the “neuralnet” command to build a two hidden layer neural network with 20 and 10 neurons in the layers, respectively.

1. Which model performed the best as a classifier on the spam data? Rhetorical question, do you evaluate this on the training or test data set? Provide insight as to why you believe that model performed best. Are there any variables that can be seen as standing out as highly predictive of spam email?
2. How many parameters (or model degrees of freedom) are in each of the above models (parts 3, 4, 5, 6, 7 and 8)? What does this have to do with model fit? How does this relate to the bias-variance tradeoff?

***R Script for Question 6***

#

# Copy training and test data to a new data set first

#

dat.train6 <- dat.train

dat.test6 <- ndat.test

dat.train6$spam <- as.factor(dat.train6$spam)

dat.test6$spam <- as.factor(dat.test6$spam)

#

# Tune the support vector machine with different penalty values

# c = 0.001 to 100

#

set.seed(123789)

tune.out.linear <- tune(svm, spam~., data = dat.train6,

kernel = "linear",

ranges = list(cost = c(0.001, 0.01, 0.1,

1, 5, 10, 100)))

summary(tune.out.linear)

#

# Choose the best model, that is the model with the value of

# “c” (penalty) that had the lowest classification error

#

svm.linear <- tune.out.linear$best.model

summary(svm.linear)

#

# Make predictions on the test set and compute classification

# errors

#

svm.pred.test.lin = predict(svm.linear, ndat.test)

table(truth = dat.test6$spam, predicted = svm.pred.test.lin)

err.svm.lin <- mean(dat.test6$spam != svm.pred.test,lin)

err.svm.lin

***R Script for Question 7 Hint 3***

#

# The first command copies the names of the predictor or independent

# variables into an array

# The second command collapses them into a single text object.

# The third command sets up the formula to run the neural net

#

predvars <- colnames(dat.train[1:57])

predvars.form <- paste(predvars, collapse = "+")

form <- as.formula(paste("spam~",predvars.form))

form

#

***R Script for Question 7 Hint 4***

#

# To build a neural net classifier with 1 hidden layer that

# has 10 neurons

#

nn.out <- neuralnet(form, dat.train, hidden = 10,

linear.output = FALSE,

lifesign = "minimal")

#

# Here are two ways of looking at the results  
#

nn.out

plot(nn.out)

#

# To use the neural net estimated on the training data

# to predict on the test data use the compute command

#

nn.test.pred.10 <- compute(nn.out, dat.test[,1:57])

#

# Assign these values to 0 or 1

#

nn.test.10 <- ifelse(nn.test.pred.10$net.result > 0.5, 1, 0)

table(truth = dat.test$spam, predicted = nn.test.10)

err.nn10 <- mean(dat.test$spam != nn.test.10)

err.nn10