**Homework 10 – Prediction with Support Vector Machines**

Support vector machines (SVM) are a highly flexible and powerful method of doing supervised machine learning. Supervised learning means that there is a criterion one is trying to predict. The typical strategy is to divide data into a training set and a test set (for example, two-thirds training and one-third test), train the model on the training set, and then see how well the model does on the test set.

In this homework, we will use a chunk of a hotel survey dataset to do some classification with an SVM. This survey dataset is the same as in HW 8. Our goal is to predict overall customer satisfaction. To simplify, we just want to predict if a customer will be happy, which we define as an overall customer satisfaction of 8 or higher.

*Hint*: As a reminder, you will need to install( ) and library( ) kernlab, to access ksvm.

**Part A: Load and condition the data**

1. The data is available on blackboard (hotelSurveyBarriot), as a JSON file.  
   *Hint: Don’t forget to use setwd() to make sure that R is looking in the right folder for your text file.*

**Part B: Create a happy customer variable**

1. To focus on predicting happy customers, we need to generate a new column (where overallCustSat is 8 or higher).

**Part C: Create training and test data sets**

Using techniques discussed in class, create two datasets – one for training, one for testing.

1. Pages 235 - 237 of the book describe how to create a training data set and a test data set. Following the strategy in the book, the training data should contain about two thirds of the whole data set, with the remaining one third going to the test data.
2. Use the *dim( )* function to demonstrate that the resulting training data set and test data set contain the appropriate number of cases.

**Step C: Build a Model using *ksvm( )***

1. Build a support vector model using the *ksvm( )* function using two or three of the variables to predict a happy customer. Once you have specified the model statement and the name of the training data set, you can use the same parameters as shown on page 237: *kernel= "rbfdot", kpar = "automatic", C = 5, cross = 3, prob.model = TRUE*
2. Write a block comment that summarizes what you learned from the book about those parameters. The two parameters of greatest interest are *C=5* and *cross=3*.
3. Store the output of kvsm( ) in a variable and then echo that variable to the console.

**Part D: Predict Values in the Test Data and Create a Confusion Matrix**

1. Use the *predict( )* function to validate the model against test data. Assuming that you put the output from the *ksvm( )* call into svmOutput and that your test data set is in a data frame called testData, the call would be:  
   *svmPred <- predict(svmOutput, testData, type = "votes")*
2. Now the svmPred object contains a list of votes in each of its rows. The votes are either for “happy” or “notHappy”. Review the contents of svmPred using *str( )* and *head( )*.
3. Create a confusion matrix (a 2 x 2 table) that compares the second row of svmPred to the contents of testData$happy variable.
4. Calculate an error rate based on what you see in the confusion matrix. See pages 243-244 for more information.

**Part F: Find a good prediction**

1. Repeat Parts C and D to try and improve your prediction
2. Explain, in a block comment, why it is valuable to have a “test” dataset that is separate from a “training” dataset?