M.S. in Analytics
ISyE 6501 Introduction to Analytics Modeling
HW 1
August 2019

(Please find the attached knn.txt and svm.txt files for source code and further notes)

Question 2.1
Describe a situation or problem from your job, everyday life, current events, etc., for which a classification model would be appropriate. List some (up to 5) predictors that you might use.

One situation for a classification model is the task of categorizing a phone call as spam or not. Below are some predictors that we use:

1. In my contacts or not: If the phone number calling me is not in my contacts, I assume it's more likely to be spam.
2. Time of day: If I get a call at an unusual hour, say 2.a.m, I assume it's more likely to be spam.
3. Eerily similar to my number: If the number calling me is very similar to my own (for example: matches the first 6 digits of my number), then I assume it's likely to be spam.
4. Consecutive calls: If I get consecutive calls near the time that I know a call was spam, I assume it's likely to be spam.
5. Voice message: If no voicemail is left, it could be spam but not necessarily. This is tricky, since some real callers do not leave voice messages, and some spammers do leave them (albeit in a different language from English).

Question 2.2
1. Using the support vector machine function ksvm contained in the R package kernlab, find a good classifier for this data. Show the equation of your classifier, and how well it classifies the data points in the full data set. (Don't worry about test/validation data yet; we'll cover that topic soon.)

The equation of our classifier is:
-0.0010065348 a1 + -0.0011729048 a2 + -0.0016261967 a3 + 0.0030064203 a4 + 1.0049405641 a5 + -0.0028259432 a6 + 0.0002600295 a7 + -0.0005349551 a8 + -0.0012283758 a9 + 0.1063633995 a10 + 0.08158492 = 0

This predicted with 86.4% accuracy. We tried multiple values for c (1, 10, 100), and found that there was no meaningful impact to the coefficients or the accuracy.

2. You are welcome, but not required, to try other (nonlinear) kernels as well; we're not covering them in this course, but they can sometimes be useful and might provide better predictions than vanilladot.

Instead of vanilladot, we tried other kernels. Below are the results.
tanhdot: 72.2%
polydot: 86.4%
ANOVAdot: 90.7%
rbfdot: 95.7%

3. Using the k-nearest-neighbors classification function kknn contained in the R kknn package, suggest a good value of k, and show how well it classifies that data points in the full data set. Don't forget to scale the data (scale=TRUE in kknn).

We looped through 20 different values of k, and we saw the following results for accuracy:

0.8149847 0.8149847 0.8149847 0.8149847 0.8516820 0.8455657 0.8470948 0.8486239 0.8470948 0.8501529 0.8516820 0.8532110 0.8516820 0.8516820 0.8532110 0.8516820 0.8516820 0.8516820 0.8501529 0.8501529

We found the 'best' value of k to be 12, with an accuracy of 85.3%. I also noted that the first 4 values of k show the lowest accuracy, at around 81.5%; perhaps it's simply better to use a k value of more than 4.