Homework 7

Problem 1

d is the correct answer.
See attached code. Following the instructions by using first 25 points of **in.dta** as training and the last 10 points of **in.dta** as validation, I found that I had the minimum classification error of 0 coming from when k = 6.

Problem 2

e is the correct answer.
See attached code. Basically doing the same thing as problem 1, but instead of evaluating error on the 10 points of the **in.dta**, we evaluate the out-of-sample error using **out.dta** and get a minimum error of 0.072 coming from k = 7.

Problem 3

d is the correct answer.
See attached code. Repeating problem 1 but switching the two data sets for training and validation led to the smallest out-of-sample error of 0.08 when k = 6.

Problem 4

d is the correct answer.
See attached code. Using code from problem 3, instead of evaluating the out-of-sample error on the first 25 points of **in.dta**, I evaluated the error on all of **out.dta** and got the smallest out-of-sample error of 0.192 coming from k = 6.

Problem 5

b is the correct answer.
The out-of-sample classification errors I obtained from problems 1 and 3 were 0.072, and 0.192 respectively, closest to answer choice b.

Problem 6

d is the correct answer.
See attached code. I ran a simulation over 10000 runs and found the expected value of e_1 to be 0.496 and the expected value of e_2 to be 0.501 and the expected value of e_{min} to be 0.331, closest to answer choice d.

Problem 7

c is the correct answer.
For the linear model, I would take the two points and get a y = ax + b equation. To get error, I would plug in the point that was left out and find the squared error. I would do this three times because for each run, there would be a different point left out and then get the average squared error. For the constant model, given two points, I would pick y = b where b is the midpoint of the two y-values of the two points chosen. Likewise, the same process of getting average error was applied. Plugging in all the values for ρ, choice answer c gave the average error of 0.5 for both the linear and constant model.

Problem 8

c is the correct answer.
See attached code. Using sklearn to simulate a SVM and the PLA from week 1, I found that the SVM had a lower error than the PLA 54.3% of the runs (1000) when N = 10, closest to answer choice c.

Problem 9

d is the correct answer.
See attached code. Using the same code as problem 8, but now changing N from 10 to 100 yielded that the SVM had a lower error than the PLA 60.4% of the time, closest to answer choice d.

Problem 10

b is the correct answer.
See attached code. Keeping track of the number of support vectors per run (len(clf.support_vectors_))and then averaging it over all 1000 runs, we got the average number of support vectors to be 2.996, closest to answer choice b.