3.

a) For the average 0-1 prediction error with k varies from n to 1, when k is larger, the error we get would be higher, as the more neighbors we included for consideration, the less the accuracy will be. And when k = 1, the prediction error would be 0, since when k = 1, it means the algorithm is just considering the data point only, and using a data point to predict itself would always be correct.

b) When k is small, the average 0-1 prediction error would probably be low, and it should be close to 1, and when k is really large (close to n), the prediction error would be rather high. And the performance on the held-out data would be poor due to the poor generalization of the model.

c) For the computational requirements, the use of cross validation is computationally expensive and time consuming, but it’s a good way for us to obtain a good value for k that would let the model end up with a higher accuracy. So if want the accuracy to be high, then the computational time may be rather long, and it want the computational time to be short, then the accuracy might be small, it’s more like a trade off between these two.

d) A simple measure may be to use a weight value. For example, for the k/3 closest neighbors, or for the neighbors within a certain distance range, let them have a larger weight, which means they are more important in determining the class label.

e) When the input dimension is high, the kNN would requirement a large amount of time. Within the prediction process of each data point, it would have to compute the distance from the data point to every other point in the data set, and with a high input dimension, this would be a disaster as the computation process would happen once for every data point. So it’s time consuming and computationally expensive. Which is not following the idea behind Occam’s Razor, which suggests using simpler model when possible.