Team member：Zhiqi Bei， Wenhan Liu

1. Variance of a sum. Show that the variance of a sum is $var[X-Y] = var[X] + var[Y] - 2cov[X,Y]$, where $cov[X,Y]$ is the covariance between random variables $X$ and $Y$.

$$
\begin{aligned}
Var[X-Y] &= E[((X-Y) - E(X-Y))^2] \\
&= E[((X-Y) - (E(X) - E(Y)))^2] \\
&= E[(X - E(X) + (Y - E(Y)))^2] \\
&= E[(X-E(X))^2 + (Y-E(Y))^2 - 2(X-E(X))(Y-E(Y))] \\
&= E[(X-E(X))^2] + E[(Y-E(Y))^2] - E[2(X-E(X))(Y-E(Y))] \\
&= var[X] + var[Y] - 2E[(X-E(X))(Y-E(Y))] \\
&= var[X] + var[Y] - 2cov[X,Y]
\end{aligned}
$$

2. Bayes rule for quality control. You're the foreman at a factory making ten million widgets per year. As a quality control step before shipment, you create a detector that tests for defective widgets before sending them to customers. The test is uniformly 95% accurate, meaning that the probability of testing positive given that the widget is defective is 0.95, as is the probability of testing negative given that the widget is not defective. Further, only one in 100,000 widgets is actually defective.

   (a) Suppose the test shows that a widget is defective. What are the chances that it's actually defective given the test result?

   (b) If we throw out all widgets that are defective, how many good widgets are thrown away per year? How many bad widgets are still shipped to customers each year?

(a)

let '+' denoted testing positive and '−' denoted testing negative.

Therefore, base on the question, we can get:

$P(+ \mid \text{defective}) = 0.95 \Rightarrow P(- \mid \text{defective}) = 0.05$

$P(- \mid \text{not defective}) = 0.05 \Rightarrow P(+ \mid \text{not defective}) = 0.95$

$P(\text{defective}) = \frac{1}{100000} \Rightarrow P(\text{not defective}) = \frac{99999}{100000}$

Then we need to find $P(\text{defective} \mid +)$, by the Bayes' theorem

$$P(\text{defective} \mid +) = \frac{P(+ \mid \text{defective})\, P(\text{defective})}{P(+)}$$

$$= \frac{P(+ \mid \text{defective})\, P(\text{defective})}{P(+ \mid \text{defective})\, P(\text{defective}) + P(+ \mid \text{not defective})\, P(\text{not defective})}$$

$$= \frac{0.95 \times \frac{1}{100000}}{0.95 \times \frac{1}{100000} + 0.05 \times \frac{99999}{100000}}$$

$$= 1.8997 \times 10^{-4}$$

(b)

making ten million widgets per year:

The number of defective is $10000000 \times \frac{1}{100000} = 100$

The number of not defective is $10000000 - 100 = 9999900$

From the last question, we know $P(+ \mid \text{defective}) = 0.95$ and $P(- \mid \text{not defective}) = 0.05$.

Then:

The 100 defective will have $100 \times 0.05 = 5$ will test negative.

The 9999900 not defective will have $9999900 \times 0.95 = 499995$ will test positive.

Therefore, 499995 good widgets are thrown away and 5 bad widgets are still ship to customers each year.

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.