The Bias-Variance Tradeoff for KNN

$$E\left[(y-\hat{f}(x))^2\right] = Var(\hat{f}(x)) + \left[E(f(x)-\hat{f}(x))\right]^2 + Var(\epsilon)$$

Variance

The variance term captures the degree to which the model, $\hat{f}(x)$ will vary from one data set to the next. This term can be better understood by replacing $\hat{f}(x)$ with the expression for the KNN model and use the properties of variance to simplify.

$$\begin{aligned}
Var(\hat{f}(x)) &= Var(\frac{1}{k}\sum_{x_i \in \mathcal{N}(x)} y_i) \\
&= \frac{1}{k^2}\sum_{x_i \in \mathcal{N}(x)} Var(y_i) \\
&= \frac{1}{k^2}\sum_{x_i \in \mathcal{N}(x)} \sigma^2 \\
&= \frac{1}{k^2} k\sigma^2 \\
&= \frac{\sigma^2}{k}
\end{aligned}$$

Line 3 is true because $Var(y_i) = Var(f+\epsilon) = Var(f) + Var(\epsilon) = 0 + \sigma^2$. This final expression looks familiar: it's the variance of a sample mean. In the KNN model, that's exactly what $\hat{f}(x)$ is: a sample mean of k y_i values.

We can write the variance as a function of k (and σ^2) as follows.

```
var_term <- function(k, sigma) {
  sigma^2 / k
}
```

Bias

The bias is the amount by which the expected fitted model diverges from the true model. To better understand this term, we start by substituting in the form of the KNN model and apply the properties of expected value.

$$\begin{aligned}
E(f(x)-\hat{f}(x)) &= E(f(x)) - E(\hat{f}(x)) \\
&= f(x) - E(\frac{1}{k}\sum_{x_i \in \mathcal{N}(x)} y_i) \\
&= f(x) - \frac{1}{k}\sum_{x_i \in \mathcal{N}(x)} E(y_i) \\
&= f(x) - \frac{1}{k}\sum_{x_i \in \mathcal{N}(x)} f(x_i)
\end{aligned}$$

This tells us that the bias at any point x is the value of f at that point minus the average of the values of the true regression function evaluated at the k points x_i in the training data that are nearest to x.

We can write this term as a function of k (and x, f, and the training x_i) as follows.

```
bias_term <- function(x, k, f, x_train) {
  abs_distances <- abs(x - x_train)
  which_k <- order(abs_distances)[1:k]
  x_train_in_N <- x_train[which_k]
  f(k) - mean(f(x_train_in_N))
}
```

The bias term is more difficult to discuss generally because it deals with the true regression function f. This is not known in practice, so the sort of analysis that we are conducting presently has to be couched as, "If the true function was *this*, then the bias-variance trade-off would look like *that*".

Putting it all together.

A single draw from X.

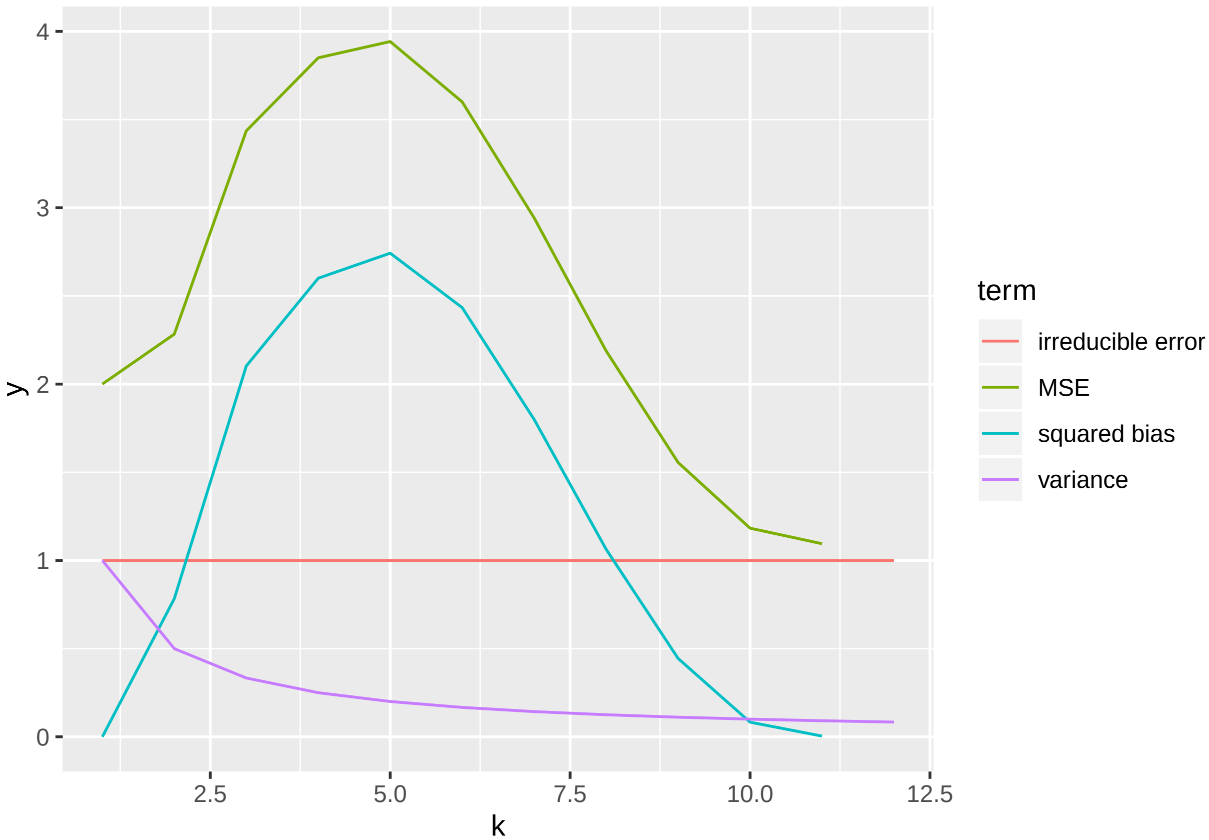
```
k <- 1:12
x <- 1.2
var_line <- rep(NA, length(k))
sq_bias_line <- rep(NA, length(k))

for (i in 1:length(k)) {
  sq_bias_line[i] <- bias_term(x, k[i], f = my_f, x_train)^2
  var_line[i] <- var_term(k[i], sigma)
}
y <- c(var_line,
       sq_bias_line,
       rep(sigma^2, length(k)),
       var_line + sq_bias_line + sigma^2)
term <- rep(c("variance", "squared bias",
              "irreducible error", "MSE"), each = length(k))

df <- tibble(k = rep(k, 4),
             y = y,
             term = term)

ggplot(df, aes(x = k, y = y, col = term)) +
  geom_line()
```

```
## Warning: Removed 2 rows containing missing values (geom_path).
```



Taking the expected value over many training data sets X.

```
n <- 20
it <- 100
var_vec <- rep(NA, length(k) * it)
sq_bias_vec <- rep(NA, length(k) * it)

for (j in 1:it) {
  x_train <- runif(n, min = 0, max = 13)
  #x_train <- rnorm(n, mean = 6.5, sd = sqrt(14))

  for (i in 1:length(k)) {
    sq_bias_vec[(j - 1) * length(k) + i] <- bias_term(x, k[i], f = my_f, x_train)^2
    var_vec[(j - 1) * length(k) + i]     <- var_term(k[i], sigma)
  }
}

expected_df <- tibble(var_vec,
                      sq_bias_vec,
                      k = rep(k, it)) %>%
  group_by(k) %>%
  summarize(var_line = mean(var_vec), # Take expected value
            sq_bias_line = mean(sq_bias_vec)) # Take expected value

y <- c(expected_df$var_line,
       expected_df$sq_bias_line,
       rep(sigma^2, length(k)),
       expected_df$var_line + expected_df$sq_bias_line + sigma^2)
```

```r
term <- rep(c("variance", "squared bias",
              "irreducible error", "MSE"), each = length(k))

df <- tibble(k = rep(k, 4),
             y = y,
             term = term)

ggplot(df, aes(x = k, y = y, col = term)) +
  geom_line() +
  scale_x_continuous(breaks = 0:13)
```