Solution Weekly Exercise 1

Julian Karch

2023-02-09

```
true_f <- function(x) {
  8 * sin(x)
}

gen_data <- function(n, true_f) {
  x <- runif(n, min = -3, max = 3)
  y <- true_f(x) + rnorm(n, sd = 1)
  data.frame(x = x, y = y)
}

train_sizes <- c(50, 10000)
degrees <- c(3, 15)
mses <- matrix(nrow = 2, ncol = 2)
rownames(mses) <- paste0("Train Size ", train_sizes)
colnames(mses) <- paste0("Degree ", degrees)

test_set <- gen_data(10000, true_f)
for (size_idx in 1:length(train_sizes)) {
  train_set <- gen_data(train_sizes[size_idx], true_f)
  for (deg_idx in 1:length(degrees)) {
    fitted_model <- lm(y ~ poly(x, degrees[deg_idx]), data = train_set)
    predictions <- predict(fitted_model, test_set)
    mses[size_idx, deg_idx] <- mean((predictions - test_set$y)^2)
  }
}
knitr::kable(mses)
```

	Degree 3	Degree 15
Train Size 50	1.303942	3.408111
Train Size 10000	1.205197	1.020201

```
test_mse_true <- mean((true_f(test_set$x) - test_set$y)^2)
```

The best prediction rule is the one that generated the data, `true_f`. Its test set MSE is 1.0202289, which is very close to the population value of 1. We know that it is 1, as the population MSE of the optimal model is equal to the error variance, which we set to 1.

For the large training set, the model with a degree of 15 has an estimated MSE of essentially 1 and thus predicts essentially as well as the best possible prediction rule. Therefore, we can conclude that this model is flexible enough to fit the true function and thus has almost no bias with respect to `true_f`. On the other hand, the degree 3 model fits substantially worse. Since the training size is large and the more complex model almost predicts perfectly, we deduce that this must be due to bias.

For the small training set, both models perform poorly. However, the model with a degree of 3 predicts substantially better. Based on the results from the large training set, we deduce that this is because the simpler model has lower variance, which is more important for smaller training sets than low bias.