Solution Weekly Exercise 1

Julian Karch

2023-02-09

Question 1

We see that the within covariance matrices are not equal. However, they only differ in one value (the covariance between X_1 and X_2). For the small training set, both methods could predict well, but we speculate that due to the relatively small assumption violation and the small training set size, the slight increase in bias for LDA compared to QDA is compensated by the relatively large reduction in variance. Thus, we expect LDA to perform better.

For the large training set, we have clear expectations. There, the contribution of variance is relatively small, so the unbiased QDA method should predict better.

Question 2

```r
library(MASS)
gen_data <- function(n) {
  p <- 3
  n1 <- n2 <- n / 2
  cov_1 <- diag(rep(1, p)) + 0.2
  cov_2 <- cov_1
  cov_2[1, 2] <- cov_2[2, 1] <- cov_2[1, 2] + 0.5
  x_class1 <- mvrnorm(n1, mu = rep(3, p), Sigma = cov_1)
  x_class2 <- mvrnorm(n2, mu = rep(2, p), Sigma = cov_2)
  x <- rbind(x_class1, x_class2)
  y <- rep(c(1, 2), c(n1, n2))
  df <- as.data.frame(cbind(x, y))
  names(df) <- c(paste0("x", 1:p), "y")
  return(df)
}


methods <- c(lda, qda)
sizes <- c(30, 10000)
accuracies <- matrix(0, length(methods), length(sizes))
colnames(accuracies) <- sizes
rownames(accuracies) <- c("lda", "qda")

reps <- 10^2

for (rep in 1:reps) {
  for (j in 1:length(sizes)) {
    train_data <- gen_data(sizes[j])
    test_data <- gen_data(10000)
    for (i in 1:length(methods)) {
      trained_model <- methods[[i]](y ~ ., data = train_data)
      accuracies[i, j] <- accuracies[i, j] +
        mean(predict(trained_model, test_data)$class == test_data$y)
    }
  }
}
print(accuracies / reps)
```

```
##           30    10000
## lda 0.717235 0.743977
## qda 0.700117 0.756581
```

Numbers are in line with expectations.

Question 3

Conceptually, the Bayes classifier uses the true posterior and assigns the class with the highest posterior. To compute the posterior, we need the prior and the likelihood. For the prior, the code implies that $P(Y = 1) = P(Y = 2) = 50$. For the likelihood, the code implies that $p(X = x|Y = 1) = f_1(x) = \mathcal{N}(x; \mu_1 = [3, 3, 3]^\top, \Sigma_1)$, and $p(X = x|Y = 2) = f_2(x) = \mathcal{N}(x; [2, 2, 2]^\top, \Sigma_2)$ with

$$\Sigma_1 = \begin{bmatrix} 1.2 & 0.2 & 0.2 \\ 0.2 & 1.2 & 0.2 \\ 0.2 & 0.2 & 1.2 \end{bmatrix}, \begin{bmatrix} 1.2 & \mathbf{0.7} & 0.2 \\ \mathbf{0.7} & 1.2 & 0.2 \\ 0.2 & 0.2 & 1.2 \end{bmatrix}$$

The Bayes classifier is given by inserting Equation 4.28 from the book. Specifically, the Bayes classifier uses discriminant functions $\delta_k(x)$ as defined in Equation 4.28, and assigns x to the class k for which the corresponding discriminant function returns the highest value.

BONUS: The implementation is as follows

```r
discriminant_f <- function(x, mu, Sigma, prior) {
  inv_sigma <- solve(Sigma)
  -.5 * t(x) %*% inv_sigma %*% x + t(x) %*% inv_sigma %*% mu -
    0.5 * t(mu) %*% inv_sigma %*% mu - 0.5 * log(det(Sigma)) + log(prior)
}

qda_bayes <- function(x, mu_1, mu_2, Sigma_1, Sigma_2, prior_1) {
  prior_2 <- 1 - prior_1
  discriminant_1 <- discriminant_f(x, mu_1, Sigma_1, prior_1)
  discriminant_2 <- discriminant_f(x, mu_2, Sigma_2, prior_2)
  ifelse(discriminant_1 >= discriminant_2, 1, 2)
}

library(purrr)
```

```
## Warning: package 'purrr' was built under R version 4.2.2
```

```r
p <- 3
cov_1 <- diag(rep(1, p)) + 0.2
cov_2 <- cov_1
cov_2[1, 2] <- cov_2[2, 1] <- cov_2[1, 2] + 0.5

# sets all variable I specify, returns a function that only has the unspecified
# variables as input (partial function evaluation)
bayes_predict_one_x <- partial(qda_bayes, mu_1 = rep(3, p), mu_2 = rep(2, p),
                                Sigma_1 = cov_1, Sigma_2 = cov_2, prior_1 = .5)
predict_all_x <- function(test_set, predict_one_x) {
  predictions <- rep(NA, nrow(test_set))
  for (i in 1:nrow(test_set)) {
    predictions[i] <- predict_one_x(t(test_set[i, -4]))
  }
  return(predictions)
}

bayes_predict_all <- partial(predict_all_x, predict_one_x = bayes_predict_one_x)
predictions <- bayes_predict_all(test_data)
mean(predictions == test_data$y)
```

```
## [1] 0.7587
```

The predictive performance of the Bayes classifier is only marginally better than that of QDA on the large training set. This is not surprising, as with such a large training set, the estimated means and covariances are close to the true values, and thus the predictions of QDA are close to the optimal predictions.