Homework 7

Zach Calhoun

3/21/2022

Problem 1

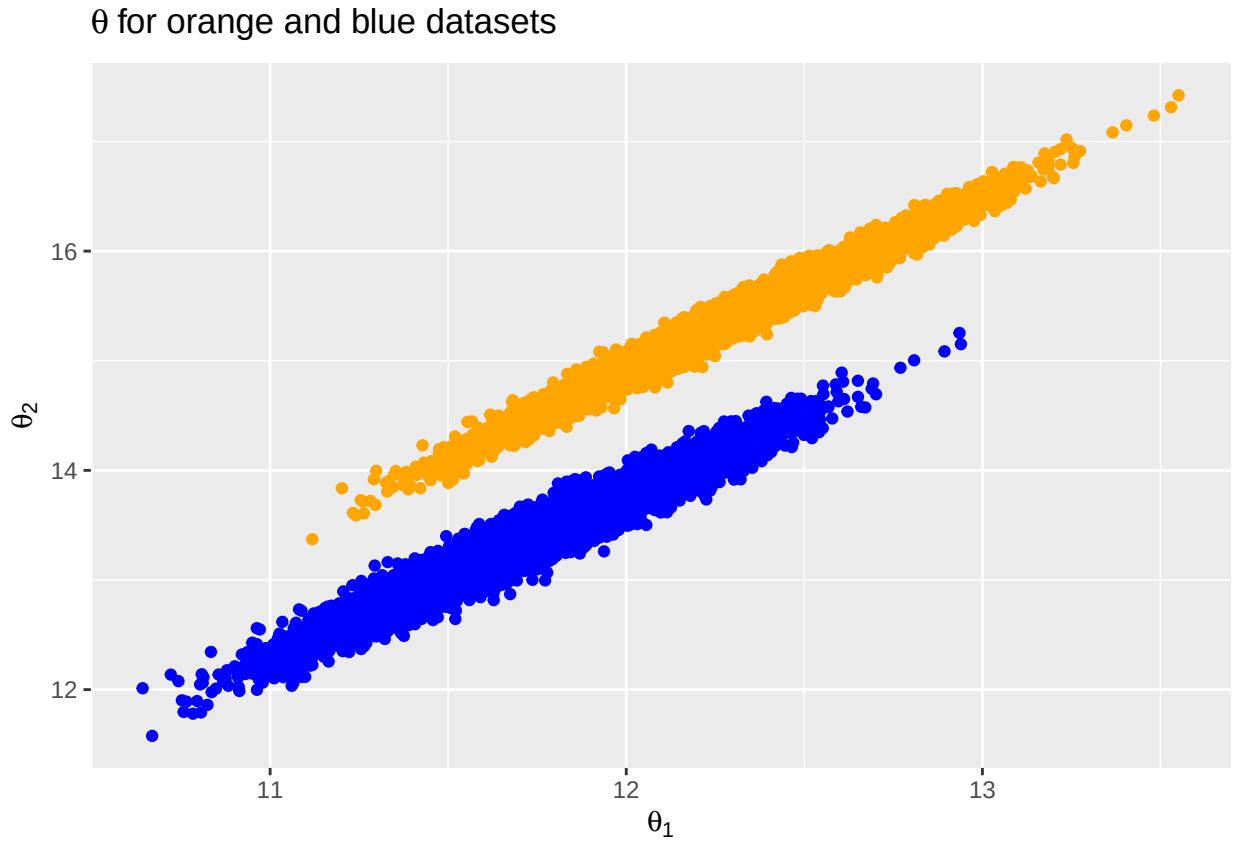
Part A

```
# Load the data
orange <- read.table('./data/orangecrab.dat')
blue <- read.table('./data/bluecrab.dat')

# This function will derive the posterior from the data. Code mainly
# copied from the lecture notes, with minor changes.
derivePosterior <- function(data, samples=10000) {

  # Set up the sample means and covariance matrices as priors
  mu0 <- apply(data, 2, mean)
  L0 <- S0 <- cov(data)
  nu0 <- 4

  THETA <- SIGMA <- NULL
  n <-dim(data)[1]
  set.seed(1)

  # Estimates of the data come from the sample mean.
  data.bar <- mu0
  # Initialize Sigma from the prior
  Sigma <- S0
  for (s in 1:samples) {
    # Update theta
    Ln <- solve(solve(L0) + n*solve(Sigma))
    mun <- Ln %*% (solve(L0) %*% mu0 + n*solve(Sigma) %*% data.bar)

    theta <- rmvnorm(1, mun, Ln)

    ## Update Sigma
    Sn <- S0 + (t(data)-c(theta)) %*% t(t(data)-c(theta))

    Sigma <- solve(rwish(nu0 + n, solve(Sn)))

    ## Save results
    THETA <- rbind(THETA, theta)
    SIGMA <- rbind(SIGMA, c(Sigma))
  }
  res <- data.frame(
```

```
    theta = THETA,
    sigma = SIGMA
  )
  return(res)
}

post.orange <- derivePosterior(orange)
post.blue <- derivePosterior(blue)
```

Part B

```
# Set up plot
ggplot() +
  geom_point(data=post.orange, aes(x=theta.1, y=theta.2), color="orange") +
  geom_point(data=post.blue, aes(x=theta.1, y=theta.2), color="blue") +
  labs( x = expression(theta[1]), y=expression(theta[2]), title=expression(theta ~ "for orange and blue
```

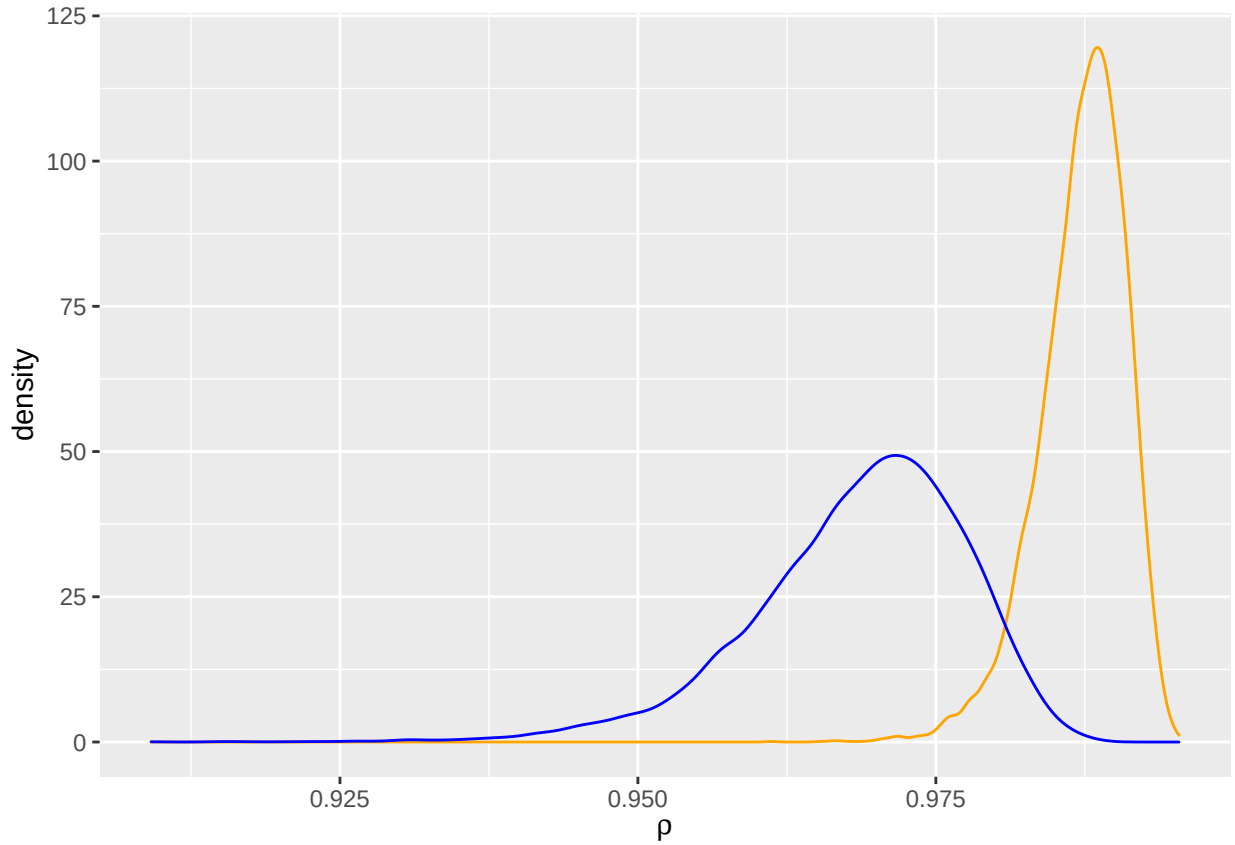


Looking at the plot above, the orange crab has a larger covariance than the blue crab data. We can tell this is the case because the width of the line is thinner. The means of both feature parameters for the orange crab are greater than the blue crab.

Part C

```
# In this cell, I'll calculate the correlation coefficients
CalcCorrelation <- function(covarianceMatrix) {
  # This function takes the covariance matrix (in this case, it is actually
  # a vector), and calculates the corresponding correlation.
  covar <- covarianceMatrix[2] # 2 or 3 is ok...

  sigma.x <- sqrt(covarianceMatrix[1])
  sigma.y <- sqrt(covarianceMatrix[4])

  return(covar/(sigma.x*sigma.y))
}

# Apply this function to the values of the array
corr <- data.frame(
    orange.corr = apply(post.orange[3:6], 1, CalcCorrelation),
    blue.corr = apply(post.blue[3:6], 1, CalcCorrelation)
  )
```

```
# Plot the correlation densities
ggplot(corr) +
  geom_density(aes(x=orange.corr), color="orange") +
  geom_density(aes(x=blue.corr), color="blue") +
  labs(x = expression(rho))
```



Based on the plot above, we can tell that the orange correlation is likely to be higher than the blue correlation.

Let's calculate the probability that the blue correlation is lower than the orange correlation. We will calculate the Monte Carlo estimate of this value as shown in the code below.

```
# Number of samples to get
samples <- 10000

output <- NULL
for (s in 1:samples) {
  # Randomly sample from the blue distribution
  blue.s <- sample(corr['blue.corr'][,1], 1)
  # Calculate the probability that we are less than the blue value
  orange.given.blue <- mean(corr['orange.corr'][,1] < blue.s)
  # Add to the output
  output <- rbind(output, orange.given.blue)
}

# Print the output
mean(output)
```

```
## [1] 0.01179191
```

Because we are trying to get $Pr(\rho_{blue} > \rho_{orange})$, we can get this value by running a Monte Carlo simulation on $P(\rho_{orange} > \rho_{blue}|\rho_{blue})Pr(\rho_{blue})$. The Monte Carlo approach estimates this probability by randomly sampling from blue, then estimating the probability that orange is less than this value using the number of orange correlation values that are below this blue value.

Looking at the output from my code ($Pr(\rho_{blue} > \rho_{orange}) \approx 0.012$), we can say that with high probability, the orange correlation is greater than the blue correlation.