# Task 2

Swaraj Borhade

2025-06-18

## Over to you! Fill in the path to your working directory.

## Example: filePath <- "C:/Users/Swaraj/Documents/Quantium_Project/"

filePath <- " " # Leave empty if .Rmd and data are in the same directory, RStudio handles it.

## If QVI_data.csv is directly in your RMD's directory:

data <- fread("QVI_data.csv")

## Ensure DATE is a Date object for lubridate functions later

data[, DATE := as.Date(DATE)]

## Set themes for plots (already in template)

theme_set(theme_bw()) theme_update(plot.title = element_text(hjust = 0.5))

#### Over to you! Add a new month ID column in the data with the format yyyymm.

## Using lubridate to extract year and month, then combine

data[, YEARMONTH := year(DATE) * 100 + month(DATE)]

#### Next, we define the measure calculations to use during the analysis.

## Over to you! For each store and month calculate total sales, number of customers,

## transactions per customer, chips per customer and the average price per unit.

## Hint: you can use uniqueN() to count distinct values in a column

measureOverTime <- data[, .( totSales = sum(TOT_SALES), nCustomers = uniqueN(L_CUSTOMER_ID), nTxnPerCust = uniqueN(TRANSACTION_ID) / uniqueN(L_CUSTOMER_ID), # Adjusted to be per unique customer nChipsPerTxn = sum(PROD_QTY) / uniqueN(TRANSACTION_ID), avgPricePerUnit = sum(TOT_SALES) / sum(PROD_QTY) ), by = .(STORE_NBR, YEARMONTH)][order(STORE_NBR, YEARMONTH)]

#### Filter to the pre-trial period and stores with full observation periods

## Pre-trial period is defined as before February 2019 (YEARMONTH < 201902)

## Full observation period means 12 months of data (12 entries for each store in measureOverTime)

storesWithFullObs <- unique(measureOverTime[, .N, by = STORE_NBR][N == 12, STORE_NBR]) preTrialMeasures <- measureOverTime[YEARMONTH < 201902 & STORE_NBR %in% storesWithFullObs]

## Define trial stores

trial_stores_list <- c(77, 86, 88)

#### Over to you! Create a function to calculate correlation for a measure, looping

#### through each control store.

#### Let's define inputTable as a metric table with potential comparison stores,

#### metricCol as the store metric used to calculate correlation on, and storeComparison

**as the store number of the trial store.** calculateCorrelation <- function(inputTable, metricCol, storeComparison) { calcCorrTable <- data.table(Store1 = numeric(), Store2 = numeric(), corr_measure = numeric())

storeNumbers <- unique(inputTable[, STORE_NBR]) # Filter out the trial store itself and other trial stores from potential controls storeNumbers <- storeNumbers[!storeNumbers %in% c(storeComparison, trial_stores_list[trial_stores_list != storeComparison])]

for (i in storeNumbers) { # Extract data for the trial store and current potential control store trial_data <- inputTable[STORE_NBR == storeComparison, get(metricCol)] control_data <- inputTable[STORE_NBR == i, get(metricCol)]

```
# Ensure there's data to correlate
if (length(trial_data) > 1 && length(control_data) > 1) {
  calculatedMeasure <- data.table(
    "Store1" = storeComparison,
    "Store2" = i,
    "corr_measure" = cor(trial_data, control_data, use = "complete.obs") # Pearson correlation
  )
  calcCorrTable <- rbind(calcCorrTable, calculatedMeasure)
}
```

} return(calcCorrTable) }

### Create a function to calculate a standardised magnitude distance for a

**measure, looping through each control store** calculateMagnitudeDistance <- function(inputTable, metricCol, storeComparison) { calcDistTable <- data.table(Store1 = numeric(), Store2 = numeric(), YEARMONTH = numeric(), measure = numeric())

storeNumbers <- unique(inputTable[, STORE_NBR]) # Filter out the trial store itself and other trial stores from potential controls storeNumbers <- storeNumbers[!storeNumbers %in% c(storeComparison, trial_stores_list[trial_stores_list != storeComparison])]

for (i in storeNumbers) { # Calculate absolute difference calculatedMeasure <- data.table( "Store1" = storeComparison, "Store2" = i, "YEARMONTH" = inputTable[STORE_NBR == storeComparison, YEARMONTH], "measure" = abs(inputTable[STORE_NBR == storeComparison, get(metricCol)] - inputTable[STORE_NBR == i, get(metricCol)]) ) calcDistTable <- rbind(calcDistTable, calculatedMeasure) }

#### Standardise the magnitude distance so that the measure ranges from 0 to 1 minMaxDist <- calcDistTable[, .(minDist = min(measure), maxDist = max(measure)), by = c("Store1", "YEARMONTH")]

distTable <- merge(calcDistTable, minMaxDist, by = c("Store1", "YEARMONTH")) distTable[, magnitudeMeasure := 1 - (measure - minDist) / (maxDist - minDist)]

finalDistTable <- distTable[, .(mag_measure = mean(magnitudeMeasure)), by = .(Store1, Store2)]

return(finalDistTable) }

### Create a function to calculate a standardised magnitude distance for a

**measure, looping through each control store** calculateMagnitudeDistance <- function(inputTable, metricCol, storeComparison) { calcDistTable <- data.table(Store1 = numeric(), Store2 = numeric(), YEARMONTH = numeric(), measure = numeric())

storeNumbers <- unique(inputTable[, STORE_NBR]) # Filter out the trial store itself and other trial stores from potential controls storeNumbers <- storeNumbers[!storeNumbers %in% c(storeComparison, trial_stores_list[trial_stores_list != storeComparison])]

for (i in storeNumbers) { # Calculate absolute difference calculatedMeasure <- data.table( "Store1" = storeComparison, "Store2" = i, "YEARMONTH" = inputTable[STORE_NBR == storeComparison, YEARMONTH], "measure" = abs(inputTable[STORE_NBR == storeComparison, get(metricCol)] - inputTable[STORE_NBR == i, get(metricCol)]) ) calcDistTable <- rbind(calcDistTable, calculatedMeasure) }

#### Standardise the magnitude distance so that the measure ranges from 0 to 1 minMaxDist <- calcDistTable[, .(minDist = min(measure), maxDist = max(measure)), by = c("Store1", "YEARMONTH")]

distTable <- merge(calcDistTable, minMaxDist, by = c("Store1", "YEARMONTH")) distTable[, magnitudeMeasure := 1 - (measure - minDist) / (maxDist - minDist)]

finalDistTable <- distTable[, .(mag_measure = mean(magnitudeMeasure)), by = .(Store1, Store2)]

return(finalDistTable) }

**Over to you! Create a combined score composed of correlation and magnitude, by**

**first merging the correlations table with the magnitude table.**

**Hint: A simple average on the scores would be 0.5 * corr_measure + 0.5 ***

**mag_measure** corr_weight <- 0.5

score_nSales <- merge(corr_nSales, magnitude_nSales, by = c("Store1", "Store2"))[, scoreNSales := corr_weight * corr_measure + (1 - corr_weight) * mag_measure]

score_nCustomers <- merge(corr_nCustomers, magnitude_nCustomers, by = c("Store1", "Store2"))[, scoreNCust := corr_weight * corr_measure + (1 - corr_weight) * mag_measure]

**Over to you! Combine scores across the drivers by first merging our sales**

**scores and customer scores into a single table** score_Control <- merge(score_nSales, score_nCustomers, by = c("Store1", "Store2")) score_Control[, finalControlScore := scoreNSales * 0.5 + scoreNCust * 0.5]

**Select control stores based on the highest matching store (closest to 1 but**

**not the store itself, i.e. the second ranked highest store)**

**Over to you! Select the most appropriate control store for trial store 77 by**

**finding the store with the highest final score.**

# Exclude the trial store itself from being selected as a control

control_store_77 <- score_Control[Store1 == trial_store & Store2 != trial_store][order(-finalControlScore)][1, Store2] control_store_77

# Convert YEARMONTH to Date for plotting

measureOverTime[, TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH %% 100, 1, sep = "-"), "%Y-%m-%d")]

pastSales <- measureOverTime[, Store_type := ifelse(STORE_NBR == trial_store, "Trial", ifelse(STORE_NBR == control_store_77, "Control", "Other stores"))][, .(totSales = mean(totSales)), by = .(YEARMONTH, Store_type, TransactionMonth) # Mean if multiple entries for same month/type][YEARMONTH < 201903] # Filter for pre-trial period for visual check

ggplot(pastSales, aes(TransactionMonth, totSales, color = Store_type)) + geom_line(size = 1) + geom_point(size = 2) + labs(x = "Month of operation", y = "Total sales", title = paste0("Total sales by month for Trial Store", trial_store, " vs Control Store ", control_store_77)) + scale_color_manual(values = c("Control" = "blue", "Trial" = "red", "Other stores" = "grey")) + theme_minimal()

#### Over to you! Conduct visual checks on customer count trends by comparing the #### trial store to the control store and other stores.

**Hint: Look at the previous plot.** pastCustomers <- measureOverTime[, Store_type := ifelse(STORE_NBR == trial_store, "Trial", ifelse(STORE_NBR == control_store_77, "Control", "Other stores"))][, .(nCustomers = mean(nCustomers)), by = .(YEARMONTH, Store_type, TransactionMonth)][YEARMONTH < 201903] # Filter for pre-trial period

ggplot(pastCustomers, aes(TransactionMonth, nCustomers, color = Store_type)) + geom_line(size = 1) + geom_point(size = 2) + labs(x = "Month of operation", y = "Total customers", title = paste0("Total customers by month for Trial Store", trial_store, " vs Control Store ", control_store_77)) + scale_color_manual(values = c("Control" = "blue", "Trial" = "red", "Other stores" = "grey")) + theme_minimal()

#### Scale pre-trial control sales to match pre-trial trial store sales

scalingFactorForControlSales <- preTrialMeasures[STORE_NBR == trial_store & YEARMONTH < 201902, sum(totSales)] / preTrialMeasures[STORE_NBR == control_store_77 & YEARMONTH < 201902, sum(totSales)]

#### Apply the scaling factor

# Make a copy to avoid modifying original measureOverTime directly for scaling

measureOverTimeSales_scaled <- copy(measureOverTime)

measureOverTimeSales_scaled[STORE_NBR == control_store_77, controlSales := totSales * scalingFactorForControlSales]

# Prepare data for percentage difference calculation for sales

trial_sales_data <- measureOverTimeSales_scaled[STORE_NBR == trial_store, .(YEARMONTH, totSales_trial = totSales)] control_sales_data <- measureOverTimeSales_scaled[STORE_NBR == control_store_77, .(YEARMONTH, controlSales)]

### Over to you! Calculate the percentage difference between scaled control sales

**and trial sales** percentageDiffSales <- merge(trial_sales_data, control_sales_data, by = "YEARMONTH")[, percentageDiff := abs(totSales_trial - controlSales) / controlSales]

# Print percentage difference for sales (optional, for checking)

# print(percentageDiffSales)

### As our null hypothesis is that the trial period is the same as the pre-trial

### period, let's take the standard deviation based on the scaled percentage difference

**in the pre-trial period** stdDevSales <- sd(percentageDiffSales[YEARMONTH < 201902, percentageDiff])

### Note that there are 8 months in the pre-trial period

**hence 8-1=7 degrees of freedom** degreesOfFreedom <- 7

### We will test with a null hypothesis of there being no difference between trial

### and control stores.

### Over to you! Calculate the t-values for the trial months. After that, find the

### 95th percentile of the t distribution with the appropriate degrees of freedom

### to check whether the hypothesis is statistically significant.

### Hint: The test statistic here is (xu)/standard deviation

# Mean percentage difference in pre-trial period (mu)

mu_sales <- mean(percentageDiffSales[YEARMONTH < 201902, percentageDiff])

# Calculate t-value for each trial month

percentageDiffSales[, tValue := (percentageDiff - mu_sales) / stdDevSales]

# Calculate 95th percentile of the t-distribution for comparison

t_critical_sales <- qt(0.95, df = degreesOfFreedom)

message(paste("T-critical (95th percentile) for sales:", round(t_critical_sales, 3))) print(percentageDiffSales[YEARMONTH >= 201902]) # Show t-values for trial months

### Trial and control store total sales

### Over to you! Create new variables Store_type, totSales and TransactionMonth in

### the data table.

# Combine data for plotting sales trends with CI

plot_data_sales <- copy(measureOverTime) # Start with a copy of measureOverTime plot_data_sales[STORE_NBR == trial_store, Store_type := "Trial"] plot_data_sales[STORE_NBR == control_store_77, Store_type := "Control"] plot_data_sales[!(STORE_NBR %in% c(trial_store, control_store_77)), Store_type := "Other stores"]

pastSales <- plot_data_sales[, .(totSales = totSales, TransactionMonth = TransactionMonth, Store_type = Store_type) # Select relevant columns][Store_type %in% c("Trial", "Control", "Other stores")] # Ensure all are included for context plot

**Control store 95th percentile** pastSales_Controls95 <- copy(pastSales[Store_type == "Control"]) pastSales_Controls95[, totSales := totSales * (1 + stdDevSales * 2)] # Apply 2 std dev for 95% approx. pastSales_Controls95[, Store_type := "Control 95th % confidence interval"]

**Control store 5th percentile** pastSales_Controls5 <- copy(pastSales[Store_type == "Control"]) pastSales_Controls5[, totSales := totSales * (1 - stdDevSales * 2)] # Apply 2 std dev for 5% approx. pastSales_Controls5[, Store_type := "Control 5th % confidence interval"]

trialAssessmentSales <- rbind(pastSales, pastSales_Controls95, pastSales_Controls5)

**Plotting these in one nice graph** ggplot(trialAssessmentSales, aes(TransactionMonth, totSales, color = Store_type)) + # Highlight trial period geom_rect(data = trialAssessmentSales[YEARMONTH >= 201902 & YEARMONTH <= 201904 & Store_type == "Trial", .(xmin = min(TransactionMonth), xmax = max(TransactionMonth), ymin = -Inf, ymax = Inf)], aes(xmin = xmin, xmax = xmax, ymin = ymin, ymax = ymax), fill = "lightblue", alpha = 0.3, color = NA, show.legend = FALSE) + geom_line(aes(linetype = Store_type), size = 1) + geom_point(size = 2) + labs(x = "Month of operation", y = "Total sales ($)", title = paste0("Total Sales: Trial Store", trial_store, " vs Control Store ", control_store_77," with 90% CI")) + scale_color_manual(values = c("Trial" = "red", "Control" = "blue", "Control 95th % confidence interval" = "darkgreen", "Control 5th % confidence interval" = "darkgreen", "Other stores" = "grey")) + scale_linetype_manual(values = c("Trial" = "solid", "Control" = "solid", "Control 95th % confidence interval” = “dashed”, “Control 5th % confidence interval” = “dashed”, “Other stores” = “dotted”)) + theme_minimal()

#### This would be a repeat of the steps before for total sales

#### Scale pre-trial control customers to match pre-trial trial store customers

#### Over to you! Compute a scaling factor to align control store customer counts

#### to our trial store.
scalingFactorForControlCust <- preTrialMeasures[STORE_NBR == trial_store & YEARMONTH < 201902, sum(nCustomers)] / preTrialMeasures[STORE_NBR == control_store_77 & YEARMONTH < 201902, sum(nCustomers)]

#### Then, apply the scaling factor to control store customer counts.

# Make a copy to avoid modifying original measureOverTime directly for scaling

measureOverTimeCusts_scaled <- copy(measureOverTime)

measureOverTimeCusts_scaled[STORE_NBR == control_store_77, controlCustomers := nCustomers * scalingFactorForControlCust]

# Prepare data for percentage difference calculation for customers

trial_cust_data <- measureOverTimeCusts_scaled[STORE_NBR == trial_store, .(YEARMONTH, nCustomers_trial = nCustomers)] control_cust_data <- measureOverTimeCusts_scaled[STORE_NBR == control_store_77, .(YEARMONTH, controlCustomers)]

#### Finally, calculate the percentage difference between scaled control store

#### customers and trial customers.
percentageDiffCust <- merge(trial_cust_data, control_cust_data, by = “YEARMONTH”)[, percentageDiff := abs(nCustomers_trial - controlCustomers) / controlCustomers]

# Print percentage difference for customers (optional, for checking)

# print(percentageDiffCust)

#### As our null hypothesis is that the trial period is the same as the pre-trial

#### period, let’s take the standard deviation based on the scaled percentage difference

#### in the pre-trial period
stdDevCust <- sd(percentageDiffCust[YEARMONTH < 201902, percentageDiff]) degreesOfFreedom <- 7 # Assuming 8 months in pre-trial, 8-1 = 7 degrees of freedom

**Trial and control store number of customers**

# Combine data for plotting customer trends with CI

plot_data_cust <- copy(measureOverTime) # Start with a copy of measureOverTime plot_data_cust[STORE_NBR == trial_store, Store_type := "Trial"] plot_data_cust[STORE_NBR == control_store_77, Store_type := "Control"] plot_data_cust[!(STORE_NBR %in% c(trial_store, control_store_77)), Store_type := "Other stores"]

pastCustomers <- plot_data_cust[, .(nCusts = nCustomers, TransactionMonth = TransactionMonth, Store_type = Store_type)][Store_type %in% c("Trial", "Control", "Other stores")]

**Control store 95th percentile** pastCustomers_Controls95 <- copy(pastCustomers[Store_type == "Control"]) pastCustomers_Controls95[, nCusts := nCusts * (1 + stdDevCust * 2)] pastCustomers_Controls95[, Store_type := "Control 95th % confidence interval"]

**Control store 5th percentile** pastCustomers_Controls5 <- copy(pastCustomers[Store_type == "Control"]) pastCustomers_Controls5[, nCusts := nCusts * (1 - stdDevCust * 2)] pastCustomers_Controls5[, Store_type := "Control 5th % confidence interval"]

trialAssessmentCust <- rbind(pastCustomers, pastCustomers_Controls95, pastCustomers_Controls5)

**Plotting these in one nice graph** ggplot(trialAssessmentCust, aes(TransactionMonth, nCusts, color = Store_type)) + # Highlight trial period geom_rect(data = trialAssessmentCust[YEARMONTH >= 201902 & YEARMONTH <= 201904 & Store_type == "Trial", .(xmin = min(TransactionMonth), xmax = max(TransactionMonth), ymin = -Inf, ymax = Inf)], aes(xmin = xmin, xmax = xmax, ymin = ymin, ymax = ymax), fill = "lightblue", alpha = 0.3, color = NA, show.legend = FALSE) + geom_line(aes(linetype = Store_type), size = 1) + geom_point(size = 2) + labs(x = "Month of operation", y = "Total number of customers", title = paste0("Total Customers: Trial Store", trial_store, " vs Control Store ", control_store_77," with 90% CI")) + scale_color_manual(values = c("Trial" = "red", "Control" = "blue", "Control 95th % confidence interval" = "darkgreen", "Control 5th % confidence interval" = "darkgreen", "Other stores" = "grey")) + scale_linetype_manual(values = c("Trial" = "solid", "Control" = "solid", "Control 95th % confidence interval" = "dashed", "Control 5th % confidence interval" = "dashed", "Other stores" = "dotted")) + theme_minimal()