Quantium Virtual Internship - Retail Strategy and Analytics - Task 2

Solution template for Task 2

This file is a solution template for the Task 2 of the Quantium Virtual Internship. It will walk you through the analysis, providing the scaffolding for your solution with gaps left for you to fill in yourself. Look for comments that say "over to you" for places where you need to add your own code! Often, there will be hints about what to do or what function to use in the text leading up to a code block - if you need a bit of extra help on how to use a function, the internet has many excellent resources on R coding, which you can find using your favourite search engine. ## Load required libraries and datasets Note that you will need to install these libraries if you have never used these before.

Point the filePath to where you have downloaded the datasets to and

```r
# Over to you! Fill in the path to your working directory. If you are on a Windows
↪  machine, you will need to use forward slashes (/) instead of backshashes (\)
filePath <- "/Users/huilinng/Desktop/Forage/Quantum - Data Analytics/Task 1/"
data <- fread(paste0(filePath,"QVI_data.csv"))
#### Set themes for plots
theme_set(theme_bw())
theme_update(plot.title = element_text(hjust = 0.5))
```

assign the data files to data.tables

Select control stores

The client has selected store numbers 77, 86 and 88 as trial stores and want control stores to be established stores that are operational for the entire observation period.

We would want to match trial stores to control stores that are similar to the trial store prior to the trial period of Feb 2019 in terms of : - Monthly overall sales revenue - Monthly number of customers - Monthly number of transactions per customer Let's first create the metrics of interest and filter to stores that are present throughout the pre-trial period.

```r
#### Calculate these measures over time for each store
#### Over to you! Add a new month ID column in the data with the format yyyymm.
data[, YEARMONTH := format(as.Date(DATE), "%Y%m")]
#### 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
```

```r
measureOverTime <- data[, .(totSales = sum(TOT_SALES),
 nCustomers = uniqueN(LYLTY_CARD_NBR),
 nTxnPerCust = .N/ uniqueN(LYLTY_CARD_NBR),
 nChipsPerTxn = sum(PROD_QTY)/ uniqueN(TXN_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
storesWithFullObs <- unique(measureOverTime[, .N, STORE_NBR][N == 12, STORE_NBR])
preTrialMeasures <- measureOverTime[YEARMONTH < 201902 & STORE_NBR %in%
storesWithFullObs, ]
```

Now we need to work out a way of ranking how similar each potential control store is to the trial store. We can calculate how correlated the performance of each store is to the trial store. Let's write a function for this so that we don't have to calculate this for each trial store and control store pair.

```r
#### 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) {
  # Convert metricCol to a character string
  metricColName <- as.character(substitute(metricCol))

  # Initialize an empty data.table for results
  calcCorrTable <- data.table(Store1 = numeric(), Store2 = numeric(), corr_measure =
↪  numeric())

  # Get the list of all store numbers, excluding the trial store
  storeNumbers <- unique(inputTable$STORE_NBR)
  storeNumbers <- storeNumbers[storeNumbers != storeComparison]

  # Check if the trial store has data
  trialStoreData <- inputTable[STORE_NBR == storeComparison, get(metricColName)]
  if (length(trialStoreData) == 0) {
    warning("No data for the trial store")
    return(calcCorrTable)  # Return empty table if no data for trial store
  }

  for (i in storeNumbers) {
    # Extract the metric values for the comparison store
    comparisonStoreData <- inputTable[STORE_NBR == i, get(metricColName)]

    if (length(comparisonStoreData) > 0) {
      # Calculate correlation
      corr_value <- cor(trialStoreData, comparisonStoreData, use = "complete.obs")

      # Add results to the table
      calculatedMeasure <- data.table(Store1 = storeComparison,
                                      Store2 = i,
                                      corr_measure = corr_value)
      calcCorrTable <- rbind(calcCorrTable, calculatedMeasure)
    }
  }

  # Print debugging information
  print(paste("Number of correlations calculated:", nrow(calcCorrTable)))

  return(calcCorrTable)
}
```

Apart from correlation, we can also calculate a standardised metric based on the absolute difference between the trial store's performance and each control store's performance. Let's write a function for this.

```
#### Create a function to calculate a standardised magnitude distance for a measure,
#### looping through each control store
calculateMagnitudeDistance <- function(inputTable, metricCol, storeComparison) {
  # Convert metricCol to a character string
  metricColName <- as.character(substitute(metricCol))

  # Initialize an empty data.table for results
  calcDistTable <- data.table(Store1 = numeric(), Store2 = numeric(), YEARMONTH =
↪  numeric(), measure = numeric())

  # Get unique store numbers
  storeNumbers <- unique(inputTable$STORE_NBR)

  # Loop through each store to calculate magnitude distance
  for (i in storeNumbers) {
    if (i != storeComparison) {  # Exclude the trial store itself
      # Extract data for the comparison store
      comparisonStoreData <- inputTable[STORE_NBR == i, get(metricColName)]

      # Calculate the magnitude distance for each month
      calculatedMeasure <- data.table(
        Store1 = storeComparison,
        Store2 = i,
        YEARMONTH = inputTable[STORE_NBR == storeComparison, YEARMONTH],
        measure = abs(inputTable[STORE_NBR == storeComparison, get(metricColName)] -
        ↪  comparisonStoreData)
      )

      # Append the result to the table
      calcDistTable <- rbind(calcDistTable, calculatedMeasure)
    }
  }

  # Print debugging information
  print(paste("Number of magnitude distances calculated:", nrow(calcDistTable)))

  # Standardize the magnitude distance
  minMaxDist <- calcDistTable[, .(minDist = min(measure, na.rm = TRUE), maxDist =
↪  max(measure, na.rm = TRUE)), by = c("Store1", "YEARMONTH")]
  distTable <- merge(calcDistTable, minMaxDist, by = c("Store1", "YEARMONTH"))

  # Calculate standardized magnitude distance
  distTable[, magnitudeMeasure := 1 - (measure - minDist) / (maxDist - minDist)]

  # Calculate the mean magnitude measure for each store pair
  finalDistTable <- distTable[, .(mag_measure = mean(magnitudeMeasure, na.rm = TRUE)), by
↪  = .(Store1, Store2)]

  return(finalDistTable)
}
```

Now let's use the functions to find the control stores! We'll select control stores based on how similar monthly total sales in dollar amounts and monthly number of customers are to the trial stores. So we will need to use our functions to get four scores, two for each of total sales and total customers.

```
#### Over to you! Use the function you created to calculate correlations against store 77
↪  using total sales and number of customers.
#### Hint: Refer back to the input names of the functions we created.
trial_store <- 77

# Calculate correlations
corr_nSales <- calculateCorrelation(preTrialMeasures, totSales, trial_store)
```

```
## [1] "Number of correlations calculated: 258"
```

```
corr_nCustomers <- calculateCorrelation(preTrialMeasures, nCustomers, trial_store)
```

```
## [1] "Number of correlations calculated: 258"
```

```
# Calculate magnitude distances
magnitude_nSales <- calculateMagnitudeDistance(preTrialMeasures, totSales, trial_store)
```

```
## [1] "Number of magnitude distances calculated: 1806"
```

```
magnitude_nCustomers <- calculateMagnitudeDistance(preTrialMeasures, nCustomers,
↪  trial_store)
```

```
## [1] "Number of magnitude distances calculated: 1806"
```

We'll need to combine the all the scores calculated using our function to create a composite score to rank on. Let's take a simple average of the correlation and magnitude scores for each driver. Note that if we consider it more important for the trend of the drivers to be similar, we can increase the weight of the correlation score (a simple average gives a weight of 0.5 to the corr_weight) or if we consider the absolute size of the drivers to be more important, we can lower the weight of the correlation score.

```
#### 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
mag_weight <- 1 - corr_weight
```

```r
score_nSales <- merge(corr_nSales, magnitude_nSales, by = c("Store1", "Store2"))
score_nSales[, scoreNSales := corr_weight * corr_measure + mag_weight * mag_measure]
score_nCustomers <- merge(corr_nCustomers, magnitude_nCustomers, by = c("Store1",
↪  "Store2"))
score_nCustomers[, scoreNCust := corr_weight * corr_measure + mag_weight * mag_measure]
```

Now we have a score for each of total number of sales and number of customers. Let's combine the two via a simple average.

```r
#### 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"), all.x
↪  = TRUE)
score_Control[, finalControlScore := scoreNSales * 0.5 + scoreNCust * 0.5]
```

The store with the highest score is then selected as the control store since it is most similar to the trial store.

```r
#### 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.

# Find the control store with the highest final score
best_match <- score_Control[order(-finalControlScore)][1]
control_store <- best_match$Store2
control_store
```

```
## [1] 233
```

```r
cat("corr_nSales rows:", nrow(corr_nSales), "\n")
```

```
## corr_nSales rows: 258
```

```r
cat("magnitude_nSales rows:", nrow(magnitude_nSales), "\n")
```

```
## magnitude_nSales rows: 258
```

```r
cat("corr_nCustomers rows:", nrow(corr_nCustomers), "\n")
```

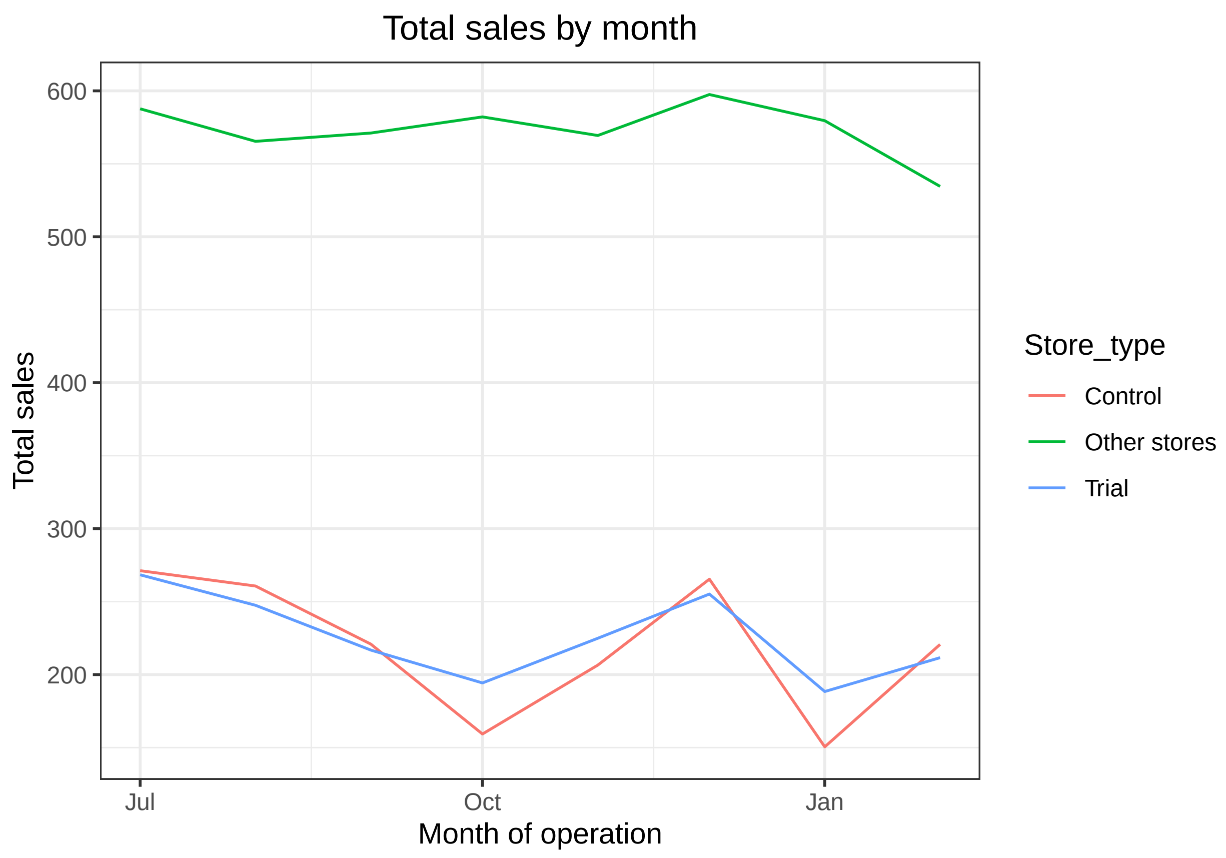
```
## corr_nCustomers rows: 258
```

```r
cat("magnitude_nCustomers rows:", nrow(magnitude_nCustomers), "\n")
```

```
## magnitude_nCustomers rows: 258
```

Now that we have found a control store, let's check visually if the drivers are indeed similar in the period before the trial. We'll look at total sales first.

```r
#### Visual checks on trends based on the drivers
measureOverTimeSales <- measureOverTime
measureOverTimeSales[, YEARMONTH := as.numeric(YEARMONTH)]
pastSales <- measureOverTimeSales[
  , Store_type := ifelse(STORE_NBR == trial_store, "Trial",
                         ifelse(STORE_NBR == control_store, "Control", "Other stores"))
][
  , totSales := mean(totSales), by = c("YEARMONTH", "Store_type")
][, TransactionMonth := as.Date(paste0(floor(YEARMONTH / 100), "-", YEARMONTH %% 100,
↪   "-01"), "%Y-%m-%d")
][YEARMONTH < 201903 , ]
ggplot(pastSales, aes(TransactionMonth, totSales, color = Store_type)) +
 geom_line() +
 labs(x = "Month of operation", y = "Total sales", title = "Total sales by month")
```



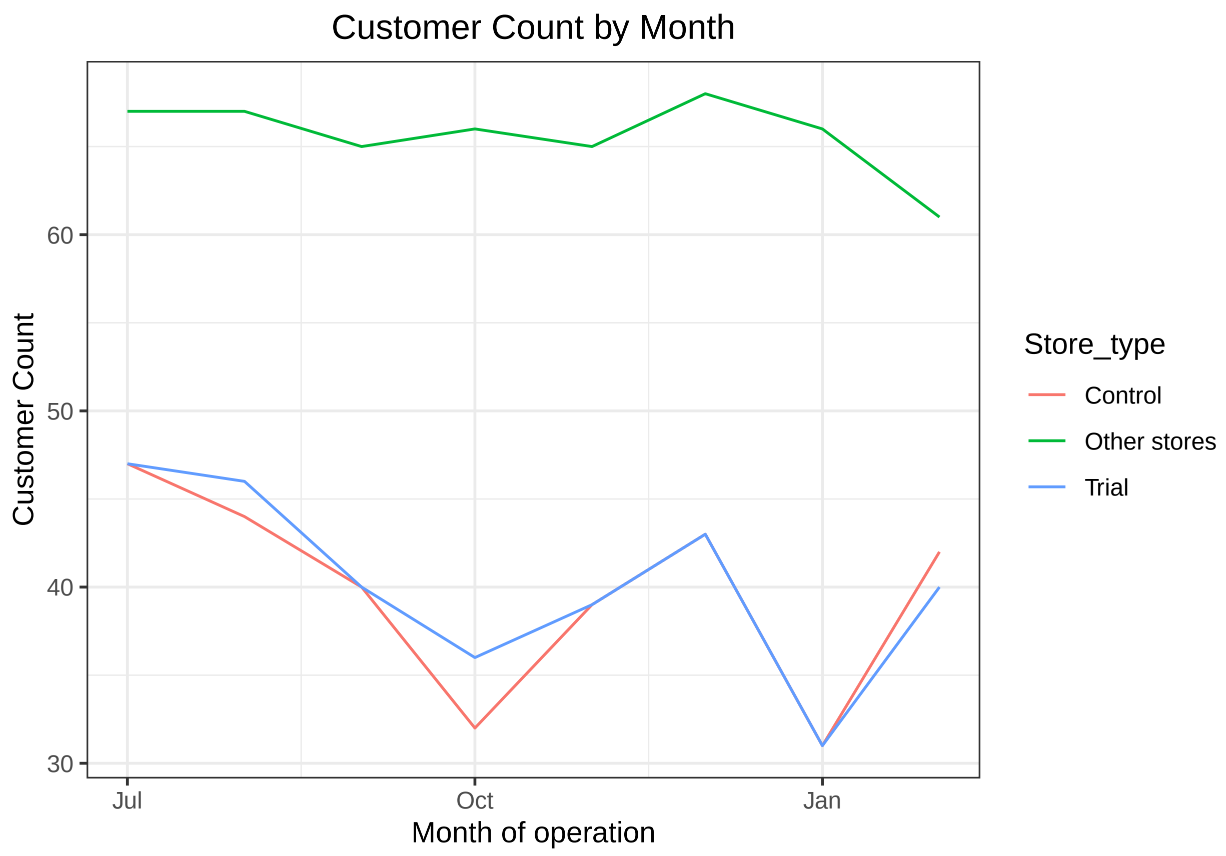
Next, number of customers.

```r
#### 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.
# Ensure measureOverTime is a data.table
measureOverTimeCusts <- as.data.table(measureOverTime)

# Convert YEARMONTH to numeric if it's not already
measureOverTimeCusts[, YEARMONTH := as.numeric(YEARMONTH)]
```

```r
# Create Store_type and calculate mean customer count
pastCustomers <- measureOverTimeCusts[
  , Store_type := ifelse(STORE_NBR == trial_store, "Trial",
                         ifelse(STORE_NBR == control_store, "Control", "Other stores"))
][
  , nCustomers := mean(nCustomers), by = c("YEARMONTH", "Store_type")
][
  , TransactionMonth := as.Date(paste0(floor(YEARMONTH / 100), "-", YEARMONTH %% 100,
↪  "-01"), "%Y-%m-%d")
][
  YEARMONTH < 201903,
]
```

```
## Warning in `[.data.table`(measureOverTimeCusts[, `:=`(Store_type,
## ifelse(STORE_NBR == : 67.202290 (type 'double') at RHS position 1
## out-of-range(NA) or truncated (precision lost) when assigning to type 'integer'
## (column 4 named 'nCustomers')
```

```r
# Plot customer counts by month
ggplot(pastCustomers, aes(TransactionMonth, nCustomers, color = Store_type)) +
  geom_line() +
  labs(x = "Month of operation", y = "Customer Count", title = "Customer Count by Month")
```



## Assessment of trial The trial period goes from the start of February 2019 to April 2019. We now want to see if there has been an uplift in overall chip sales. We'll start with scaling the control store's sales to a level similar to control for any differences between the two stores outside of the trial period.

```
#### 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 &
YEARMONTH < 201902, sum(totSales)]
#### Apply the scaling factor
measureOverTimeSales <- measureOverTime
scaledControlSales <- measureOverTimeSales[STORE_NBR == control_store, ][ ,
controlSales := totSales * scalingFactorForControlSales]
```

Now that we have comparable sales figures for the control store, we can calculate the percentage difference between the scaled control sales and the trial store's sales during the trial period.

```
#### Over to you! Calculate the percentage difference between scaled control sales and
↪  trial sales
# Filter the data for the trial period (February 2019 to April 2019)
trialPeriodSales <- measureOverTimeSales[
  YEARMONTH >= 201902 & YEARMONTH <= 201904,
  .(STORE_NBR, YEARMONTH, totSales)
]

# Extract trial store sales
trialSales <- trialPeriodSales[STORE_NBR == 77, .(YEARMONTH, trialSales = totSales)]

# Extract and scale control store sales
scaledControlSales <- trialPeriodSales[STORE_NBR == 233, .(YEARMONTH, controlSales =
↪  totSales)]

# Merge trial and scaled control sales
mergedSales <- merge(trialSales, scaledControlSales, by = "YEARMONTH")

# Calculate percentage difference
mergedSales[, percentageDiff := ((controlSales - trialSales) / trialSales) * 100]

# Print the percentage difference
print(mergedSales)
```

```
## Key: <YEARMONTH>
##    YEARMONTH trialSales controlSales percentageDiff
##        <num>      <num>        <num>          <num>
## 1:    201902      211.6        220.7       4.300567
## 2:    201903      255.1        180.6     -29.204234
## 3:    201904      258.1        144.2     -44.130182
```

Let's see if the difference is significant!

```
#### 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
stdDev <- sd(mergedSales[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 0 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 (x - u)/standard deviation
# Define the null hypothesis mean (0 difference)
nullMean <- 0
# Calculate t-values
mergedSales[, tValue := (percentageDiff - nullMean) / stdDev]

# Convert YEARMONTH to TransactionMonth for plotting
mergedSales[, TransactionMonth := as.Date(paste0(floor(YEARMONTH / 100), "-", YEARMONTH
↪  %% 100, "-01"), "%Y-%m-%d")]

# Calculate the 95th percentile of the t-distribution
degreesOfFreedom <- 7
criticalValue <- qt(0.95, degreesOfFreedom)

# Print t-values and critical value
print(mergedSales)
```

```
## Key: <YEARMONTH>
##    YEARMONTH trialSales controlSales percentageDiff tValue TransactionMonth
##        <num>      <num>        <num>          <num>  <num>           <Date>
## 1:    201902      211.6        220.7       4.300567     NA       2019-02-01
## 2:    201903      255.1        180.6     -29.204234     NA       2019-03-01
## 3:    201904      258.1        144.2     -44.130182     NA       2019-04-01
```

```
print(paste("95th percentile of t-distribution with", degreesOfFreedom, "degrees of
↪  freedom:", criticalValue))
```

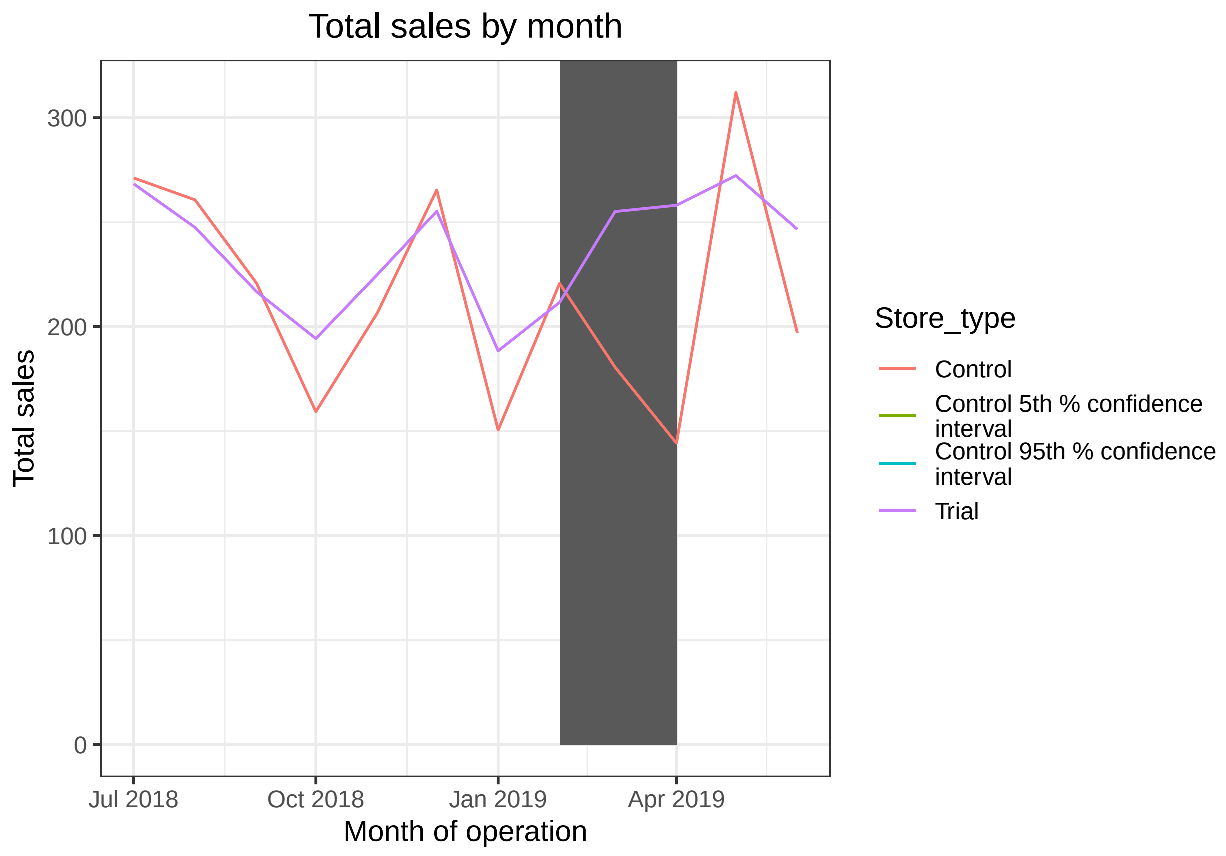
```
## [1] "95th percentile of t-distribution with 7 degrees of freedom:
1.89457860509001"
```

We can observe that the t-value is much larger than the 95th percentile value of the t-distribution for March and April - i.e. the increase in sales in the trial store in March and April is statistically greater than in the control store. Let's create a more visual version of this by plotting the sales of the control store, the sales of the trial stores and the 95th percentile value of sales of the control store.

```
measureOverTimeSales <- measureOverTime
#### Trial and control store total sales
#### Over to you! Create new variables Store_type, totSales and TransactionMonth in the
↪  data table.
pastSales <- measureOverTimeSales[
  , Store_type := ifelse(STORE_NBR == trial_store, "Trial",
                         ifelse(STORE_NBR == control_store, "Control", "Other stores"))
][, totSales := mean(totSales), by = .(YEARMONTH, Store_type)
][, TransactionMonth := as.Date(paste0(floor(YEARMONTH / 100), "-", YEARMONTH %% 100,
↪  "-01"), "%Y-%m-%d")
][Store_type %in% c("Trial", "Control")]

#### Control store 95th percentile
```

```
pastSales_Controls95 <- pastSales[Store_type == "Control",
                                  ][, totSales := totSales * (1 + stdDev * 2)
                                  ][, Store_type := "Control 95th % confidence
interval"]
#### Control store 5th percentile
pastSales_Controls5 <- pastSales[Store_type == "Control",
                                 ][, totSales := totSales * (1 - stdDev * 2)
                                 ][, Store_type := "Control 5th % confidence
interval"]
trialAssessment <- rbind(pastSales, pastSales_Controls95, pastSales_Controls5)
#### Plotting these in one nice graph
ggplot(trialAssessment, aes(TransactionMonth, totSales, color = Store_type)) +
 geom_rect(data = trialAssessment[ YEARMONTH < 201905 & YEARMONTH > 201901 ,],
aes(xmin = min(TransactionMonth), xmax = max(TransactionMonth), ymin = 0 , ymax =
Inf, color = NULL), show.legend = FALSE) +
 geom_line() +
 labs(x = "Month of operation", y = "Total sales", title = "Total sales by month")
```

```
## Warning: Removed 24 rows containing missing values or values outside the scale range
## (`geom_line()`).
```



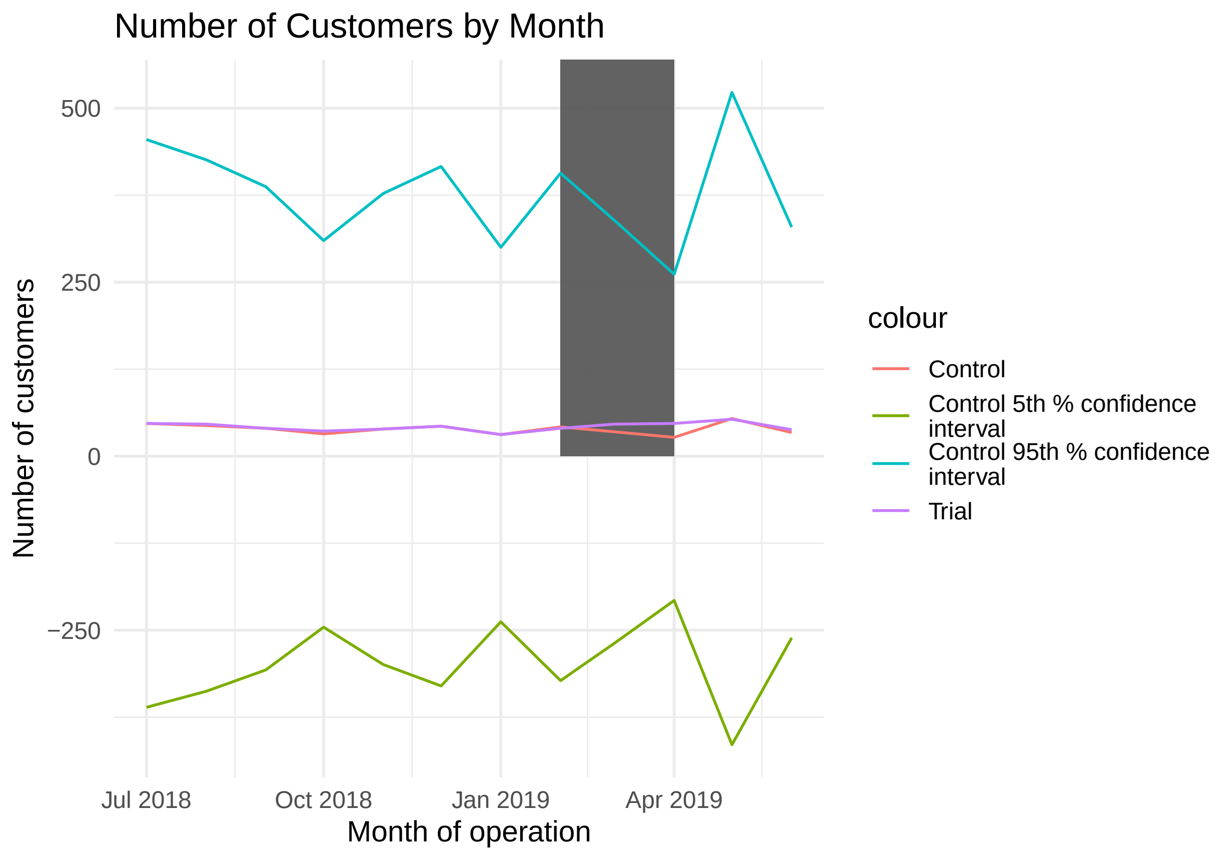
The results show that the trial in store 77 is significantly different to its control store in the trial period as the trial store performance lies outside the 5% to 95% confidence interval of the control store in two of the three trial months. Let's have a look at assessing this for number of customers as well.

```r
#### 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.
#### Then, apply the scaling factor to control store customer counts.
#### Finally, calculate the percentage difference between scaled control store customers
↪  and trial customers.
scalingFactorForControlCust <- measureOverTimeCusts[
  STORE_NBR == trial_store & YEARMONTH < 201902,
  sum(nCustomers)
] / measureOverTimeCusts[
  STORE_NBR == control_store & YEARMONTH < 201902,
  sum(nCustomers)
]
# Apply scaling factor to control store customers
scaledControlCustomers <- measureOverTimeCusts[
  STORE_NBR == control_store,
  ][, controlCustomers := nCustomers * scalingFactorForControlCust
  ][, Store_type := "Control"]

# Add trial store customer counts for comparison
trialCustomers <- measureOverTimeCusts[
  STORE_NBR == trial_store,
  ][, Store_type := "Trial"]

# Combine trial and scaled control customer counts
combinedCustomers <- rbind(trialCustomers, scaledControlCustomers, fill = TRUE)

# Calculate the percentage difference
percentageDiff <- merge(
  trialCustomers[, .(YEARMONTH, Store_type, nCustomers)],
  scaledControlCustomers[, .(YEARMONTH, Store_type, controlCustomers)],
  by = "YEARMONTH"
)[, percentageDiff := (controlCustomers - nCustomers) / nCustomers * 100]
```

Let’s again see if the difference is significant visually!

```r
#### 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
stdDev <- sd(percentageDiff[YEARMONTH < 201902 , percentageDiff])
degreesOfFreedom <- 7
#### Trial and control store number of customers
pastCustomers <- measureOverTimeCusts[, nCusts := mean(nCustomers), by =
c("YEARMONTH", "Store_type")
][Store_type %in% c("Trial", "Control"), ]
#### Control store 95th percentile
pastCustomers_Controls95 <- pastCustomers[Store_type == "Control",
                                          ][, nCusts := nCusts * (1 + stdDev * 2)
                                          ][, Store_type := "Control 95th % confidence
interval"]
#### Control store 5th percentile
pastCustomers_Controls5 <- pastCustomers[Store_type == "Control",
 ][, nCusts := nCusts * (1 - stdDev * 2)
 ][, Store_type := "Control 5th % confidence
interval"]
trialAssessment <- rbind(pastCustomers, pastCustomers_Controls95,
pastCustomers_Controls5)
#### Over to you! Plot everything into one nice graph.
#### Hint: geom_rect creates a rectangle in the plot. Use this to highlight the trial
↪  period in our graph.
ggplot() +
 geom_rect(
    data = trialAssessment[YEARMONTH < 201905 & YEARMONTH > 201901, ],
    aes(xmin = min(TransactionMonth), xmax = max(TransactionMonth), ymin = 0, ymax = Inf,
    ↪  color = NULL),
    show.legend = FALSE, alpha = 0.2
  ) +
  geom_line(data = trialAssessment, aes(x = TransactionMonth, y = nCusts, color =
  ↪  Store_type)) +
  labs(
    x = "Month of operation",
    y = "Number of customers",
    title = "Number of Customers by Month"
  ) +
  theme_minimal()
```

```
## Warning in geom_rect(data = trialAssessment[YEARMONTH < 201905 & YEARMONTH > : All aesthetics have l
## i Did you mean to use `annotate()`?
```



Let's repeat finding the control store and assessing the impact of the trial for each of the other two trial stores.

We will repeat the above codes to find that trial store 86 and 88 has control store 155 and 237.

Total number of customers in the trial period for the trial store is significantly higher than the control store for two out of three months, which indicates a positive trial effect. ## Conclusion Good work! We've found control stores 233, 155, 237 for trial stores 77, 86 and 88 respectively. The results for trial stores 77 and 88 during the trial period show a significant difference in at least two of the three trial months but this is not the case for trial store 86. We can check with the client if the implementation of the trial was different in trial store 86 but overall, the trial shows a significant increase in sales. Now that we have finished our analysis, we can prepare our presentation to the Category Manager.