```
R version 4.4.3 (2025-02-28 ucrt) -- "Trophy Case"
Copyright (C) 2025 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

>  library(data.table)
data.table 1.17.0 using 2 threads (see ?getDTthreads).  Latest news: r-datatable.com
>  library(ggplot2)
>  library(tidyr)
>  filePath <- "C:/Users/PC/OneDrive/Desktop/Projects/Quantium/QVI_data (1).csv"
>  data <- fread(paste0( "C:/Users/PC/OneDrive/Desktop/Projects/Quantium/QVI_data (1).csv"))
>  theme_set(theme_bw())
>  theme_update(plot.title = element_text(hjust = 0.5))
> # Load required packages
> library(data.table)
> library(lubridate)

Attaching package: ‘lubridate’

The following objects are masked from ‘package:data.table’:

    hour, isoweek, mday, minute, month, quarter, second, wday, week,
    yday, year

The following objects are masked from ‘package:base’:

    date, intersect, setdiff, union

>
> # Load the dataset
> data <- fread("QVI_data (1).csv")
.csv) was unexpected at this time.
Error in fread("QVI_data (1).csv") :
  File 'C:\Users\PC\AppData\Local\Temp\RtmpSsJFVE\file48c530d4ff6' does not exist or is non-reada
ble. getwd()=='C:/Users/PC/OneDrive/Documents'
In addition: Warning message:
In (if (.Platform$OS.type == "unix") system else shell)(paste0("(",  :
  '(QVI_data (1).csv) > C:\Users\PC\AppData\Local\Temp\RtmpSsJFVE\file48c530d4ff6' execution fail
ed with error code 1
>
> # Create YEARMONTH column in the format yyyymm
> data[, YEARMONTH := format(as.Date(DATE, format = "%Y-%m-%d"), "%Y%m")]
>
> # Convert YEARMONTH to numeric for easier filtering
> data[, YEARMONTH := as.numeric(YEARMONTH)]
>
> # Define the pre-trial period cutoff (before Feb 2019)
> pre_trial_cutoff <- 201902
>
> # Get list of stores that are present throughout the pre-trial period
> pre_trial_months <- unique(data[YEARMONTH < pre_trial_cutoff, YEARMONTH])
> n_months <- length(pre_trial_months)
>
> # Count months each store was active in pre-trial period
```

```
> store_month_counts <- data[YEARMONTH < pre_trial_cutoff, .N, by = .(STORE_NBR, YEARMONTH)][, .N
, by = STORE_NBR]
> full_period_stores <- store_month_counts[N == n_months, STORE_NBR]
>
> # Filter to those stores
> pre_trial_data <- data[STORE_NBR %in% full_period_stores & YEARMONTH < pre_trial_cutoff]
>
> # Calculate monthly metrics for each store
> monthly_metrics <- pre_trial_data[, .(
+   monthly_revenue = sum(TOT_SALES, na.rm = TRUE),
+   monthly_customers = uniqueN(LYLTY_CARD_NBR),
+   transactions_per_customer = .N / uniqueN(LYLTY_CARD_NBR)
+ ), by = .(STORE_NBR, YEARMONTH)]
>
> # View the result
> head(monthly_metrics)
   STORE_NBR YEARMONTH monthly_revenue monthly_customers
       <int>     <num>           <num>              <int>
1:         1    201810           188.1                 44
2:         1    201809           278.8                 59
3:         1    201811           192.6                 46
4:         1    201812           189.6                 42
5:         1    201807           206.9                 49
6:         1    201901           154.8                 35
   transactions_per_customer
                       <num>
1:                  1.022727
2:                  1.050847
3:                  1.021739
4:                  1.119048
5:                  1.061224
6:                  1.028571
>  measureOverTime <- data[, .(totSales = ,
+                                nCustomers = ,
+                                nTxnPerCust = ,
+                                nChipsPerTxn = ,
+                                avgPricePerUnit =  )
+                            , by = ][order()]
Error in list(, , , , ) : argument 1 is empty
> measureOverTime <- data[, .(
+   totSales = sum(TOT_SALES, na.rm = TRUE),
+   nCustomers = uniqueN(LYLTY_CARD_NBR),
+   nTxnPerCust = .N / uniqueN(LYLTY_CARD_NBR),
+   nChipsPerTxn = sum(PROD_QTY, na.rm = TRUE) / .N,
+   avgPricePerUnit = sum(TOT_SALES, na.rm = TRUE) / sum(PROD_QTY, na.rm = TRUE)
+ ), by = .(STORE_NBR, YEARMONTH)][order(STORE_NBR, YEARMONTH)]
>
> # View the first few rows
> head(measureOverTime)
   STORE_NBR YEARMONTH totSales nCustomers nTxnPerCust nChipsPerTxn
       <int>     <num>    <num>      <int>       <num>        <num>
1:         1    201807    206.9         49    1.061224     1.192308
2:         1    201808    176.1         42    1.023810     1.255814
3:         1    201809    278.8         59    1.050847     1.209677
4:         1    201810    188.1         44    1.022727     1.288889
5:         1    201811    192.6         46    1.021739     1.212766
6:         1    201812    189.6         42    1.119048     1.212766
   avgPricePerUnit
             <num>
1:        3.337097
2:        3.261111
3:        3.717333
4:        3.243103
5:        3.378947
6:        3.326316
>  storesWithFullObs <- unique(measureOverTime[, .N, STORE_NBR][N == 12, STORE_NBR])
>  preTrialMeasures <- measureOverTime[YEARMONTH < 201902 & STORE_NBR %in%
+ storesWithFullObs, ]
> # Load required package
> library(data.table)
```

```
>
> # Function to calculate correlations for a given measure
> calculate_correlations <- function(measure_data, trial_store_nbr, measure, control_store_nbrs)
{
+
+   # Subset for the trial store
+   trial_store_data <- measure_data[STORE_NBR == trial_store_nbr, .(YEARMONTH, trial_value = get
(measure))]
+
+   # Initialize results list
+   correlation_results <- list()
+
+   # Loop through each control store
+   for (store in control_store_nbrs) {
+
+     control_store_data <- measure_data[STORE_NBR == store, .(YEARMONTH, control_value = get(mea
sure))]
+
+     # Merge on YEARMONTH to align months
+     merged_data <- merge(trial_store_data, control_store_data, by = "YEARMONTH")
+
+     # Calculate correlation
+     corr_value <- cor(merged_data$trial_value, merged_data$control_value, use = "complete.obs")
+
+     # Store result
+     correlation_results[[as.character(store)]] <- data.table(
+       Control_Store = store,
+       Correlation = corr_value
+     )
+   }
+
+   # Combine all into one data.table
+   rbindlist(correlation_results)
+ }
>
> # Example usage:
> # control_stores <- setdiff(unique(measureOverTime$STORE_NBR), c(77, 86, 88))
> # correlations_for_77 <- calculate_correlations(measureOverTime, 77, "totSales", control_stores
)
> # head(correlations_for_77)
> calculateCorrelation <- function(inputTable, metricCol, storeComparison) {
+  calcCorrTable = data.table(Store1 = numeric(), Store2 = numeric(), corr_measure =
+ numeric())
+   storeNumbers <-
+
+   for (i in storeNumbers) {
+   calculatedMeasure = data.table("Store1" = ,
+     "Store2" = ,
+     "corr_measure" =
+     )
+
+   calcCorrTable <- rbind(calcCorrTable, calculatedMeasure)
+   }
+   return(calcCorrTable)
+  }
> calculateMagnitudeDistance <- function(inputTable, metricCol, storeComparison) {
+  calcDistTable = data.table(Store1 = numeric(), Store2 = numeric(), YEARMONTH =
+ numeric(), measure = numeric())
+   storeNumbers <- unique(inputTable[, STORE_NBR])
+
+   for (i in storeNumbers) {
+   calculatedMeasure = data.table("Store1" = storeComparison
+                                  , "Store2" = i
+                                  , "YEARMONTH" = inputTable[STORE_NBR ==
+ storeComparison, YEARMONTH]
+                                  , "measure" = abs(inputTable[STORE_NBR ==
+ storeComparison, eval(metricCol)]
+                                                    - inputTable[STORE_NBR == i,
+ eval(metricCol)])
+                                    )
```

```
+   calcDistTable <- rbind(calcDistTable, calculatedMeasure)
+   }
+  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)
+  }
> # Define trial and control stores
> trial_store <- 77
> all_trial_stores <- c(77, 86, 88)
> control_stores <- setdiff(unique(measureOverTime$STORE_NBR), all_trial_stores)
>
> # Calculate correlation scores
> cor_totSales_77 <- calculate_correlations(measureOverTime, trial_store, "totSales", control_sto
res)
Warning messages:
1: In cor(merged_data$trial_value, merged_data$control_value, use = "complete.obs") :
  the standard deviation is zero
2: In cor(merged_data$trial_value, merged_data$control_value, use = "complete.obs") :
  the standard deviation is zero
> cor_nCustomers_77 <- calculate_correlations(measureOverTime, trial_store, "nCustomers", control
_stores)
Warning messages:
1: In cor(merged_data$trial_value, merged_data$control_value, use = "complete.obs") :
  the standard deviation is zero
2: In cor(merged_data$trial_value, merged_data$control_value, use = "complete.obs") :
  the standard deviation is zero
3: In cor(merged_data$trial_value, merged_data$control_value, use = "complete.obs") :
  the standard deviation is zero
4: In cor(merged_data$trial_value, merged_data$control_value, use = "complete.obs") :
  the standard deviation is zero
5: In cor(merged_data$trial_value, merged_data$control_value, use = "complete.obs") :
  the standard deviation is zero
6: In cor(merged_data$trial_value, merged_data$control_value, use = "complete.obs") :
  the standard deviation is zero
>
> # Merge the two correlation tables
> correlation_scores_77 <- merge(cor_totSales_77, cor_nCustomers_77,
+                                by = "Control_Store", suffixes = c("_totSales", "_nCustomers"))
>
> # View top control matches sorted by combined average correlation
> correlation_scores_77[, avg_correlation := (Correlation_totSales + Correlation_nCustomers) / 2]
> correlation_scores_77 <- correlation_scores_77[order(-avg_correlation)]
>
> # Display top matches
> head(correlation_scores_77)
   Control_Store Correlation_totSales Correlation_nCustomers avg_correlation
           <int>                <num>                 <num>           <num>
1:            41            0.7622919             0.7610245       0.7616582
2:            35            0.6997078             0.7877372       0.7437225
3:           167            0.6960754             0.7487930       0.7224342
4:           233            0.6130627             0.6767367       0.6448997
5:            71            0.5346495             0.7370964       0.6358729
6:           234            0.6322040             0.6367640       0.6344840
> magnitude_nSales <- calculateMagnitudeDistance(preTrialMeasures, quote(totSales),
+ trial_store)
>  magnitude_nCustomers <- calculateMagnitudeDistance(preTrialMeasures,
+ quote(nCustomers), trial_store)
> # Set correlation weight (adjust this to emphasize trend vs size)
> corr_weight <- 0.5
>
> # Function to calculate magnitude similarity
> calculate_magnitude_similarity <- function(measure_data, trial_store_nbr, measure, control_stor
e_nbrs) {
+   trial_data <- measure_data[STORE_NBR == trial_store_nbr, .(YEARMONTH, trial_value = get(measu
re))]
```

```
+   magnitude_scores <- list()
+
+   for (store in control_store_nbrs) {
+     control_data <- measure_data[STORE_NBR == store, .(YEARMONTH, control_value = get(measure))
]
+     merged_data <- merge(trial_data, control_data, by = "YEARMONTH")
+
+     # Avoid division by zero
+     merged_data <- merged_data[trial_value != 0]
+
+     # Calculate absolute percentage difference
+     merged_data[, perc_diff := abs(trial_value - control_value) / trial_value]
+
+     # Take 1 - mean(perc_diff) as similarity score
+     similarity_score <- 1 - mean(merged_data$perc_diff, na.rm = TRUE)
+
+     magnitude_scores[[as.character(store)]] <- data.table(
+       Control_Store = store,
+       Magnitude = similarity_score
+     )
+   }
+
+   rbindlist(magnitude_scores)
+ }
>
> # Calculate magnitude similarities for store 77
> mag_totSales_77 <- calculate_magnitude_similarity(measureOverTime, 77, "totSales", control_stor
es)
> mag_nCustomers_77 <- calculate_magnitude_similarity(measureOverTime, 77, "nCustomers", control_
stores)
>
> # Combine correlation and magnitude into composite scores
> score_nSales <- merge(cor_totSales_77, mag_totSales_77, by = "Control_Store")
> score_nSales[, scoreNSales := corr_weight * Correlation + (1 - corr_weight) * Magnitude]
>
> score_nCustomers <- merge(cor_nCustomers_77, mag_nCustomers_77, by = "Control_Store")
> score_nCustomers[, scoreNCust := corr_weight * Correlation + (1 - corr_weight) * Magnitude]
>
> # Merge the two scores into a final ranking table
> final_scores_77 <- merge(score_nSales[, .(Control_Store, scoreNSales)],
+                          score_nCustomers[, .(Control_Store, scoreNCust)],
+                          by = "Control_Store")
>
> # Final composite score (simple average of the two)
> final_scores_77[, finalScore := (scoreNSales + scoreNCust) / 2]
>
> # Sort by best matching control stores
> final_scores_77 <- final_scores_77[order(-finalScore)]
>
> # View top control matches
> head(final_scores_77)
   Control_Store scoreNSales scoreNCust finalScore
           <int>       <num>      <num>      <num>
1:            41   0.8048068  0.8455908  0.8251988
2:           167   0.7260575  0.8223824  0.7742199
3:           233   0.7398817  0.8083881  0.7741349
4:            53   0.6866653  0.7368638  0.7117645
5:            35   0.6150337  0.7774132  0.6962235
6:           115   0.6292581  0.7232255  0.6762418
> # Merge scoreNSales and scoreNCust into one table
> score_Control <- merge(score_nSales[, .(Control_Store, scoreNSales)],
+                        score_nCustomers[, .(Control_Store, scoreNCust)],
+                        by = "Control_Store")
>
> # Calculate the final combined control score
> score_Control[, finalControlScore := 0.5 * scoreNSales + 0.5 * scoreNCust]
>
> # Sort descending to find best match
> score_Control <- score_Control[order(-finalControlScore)]
>
```

```
> # Select the best match (highest score)
> control_store <- score_Control[1, Control_Store]
>
> # Output result
> print(paste("Best control store for trial store 77 is:", control_store))
[1] "Best control store for trial store 77 is: 41"
>  measureOverTimeSales <- measureOverTime
>  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(paste(YEARMONTH %/%
+ 100, YEARMONTH %% 100, 1, sep = "-"), "%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")
> # Load ggplot2 for plotting
> library(ggplot2)
>
> # Define stores to compare
> stores_to_plot <- c(77, control_store)
>
> # Filter measureOverTime for customer numbers
> cust_data_plot <- measureOverTime[STORE_NBR %in% stores_to_plot, .(STORE_NBR, YEARMONTH, nCusto
mers)]
>
> # Plot
> ggplot(cust_data_plot, aes(x = YEARMONTH, y = nCustomers, color = factor(STORE_NBR))) +
+   geom_line(size = 1.1) +
+   labs(title = "Customer Count Trend: Trial vs Control Store",
+        x = "Month",
+        y = "Number of Customers",
+        color = "Store") +
+   theme_minimal() +
+   theme(axis.text.x = element_text(angle = 45, hjust = 1))
Warning message:
Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
ℹ Please use `linewidth` instead.
This warning is displayed once every 8 hours.
Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
generated.
> save.image("C:\\Users\\PC\\OneDrive\\Desktop\\python\\qt2")
> scalingFactorForControlSales <- preTrialMeasures[STORE_NBR == trial_store &
+ YEARMONTH < 201902, sum(totSales)]/preTrialMeasures[STORE_NBR == control_store &
+ YEARMONTH < 201902, sum(totSales)]
>  measureOverTimeSales <- measureOverTime
>  scaledControlSales <- measureOverTimeSales[STORE_NBR == control_store, ][ ,
+ controlSales := totSales * scalingFactorForControlSales]
> # Merge trial and scaled control sales
> percentageDiff <- merge(trial_sales[, .(YEARMONTH, trialSales = totSales)],
+                         scaled_control_sales[, .(YEARMONTH, scaledControlSales = scaledSales)],
+                         by = "YEARMONTH")
Error: object 'trial_sales' not found
>
> # Calculate percentage difference
> percentageDiff[, percentageDiff := (trialSales - scaledControlSales) / scaledControlSales]
Error: object 'percentageDiff' not found
>
> # View result
> head(percentageDiff)
Error: object 'percentageDiff' not found
> # Filter trial period: Feb 2019 to Apr 2019
> trial_period <- 201902:201904
>
> # Extract trial store sales
> trialSales <- measureOverTime[STORE_NBR == trial_store & YEARMONTH %in% trial_period,
+                               .(YEARMONTH, trialSales = totSales)]
```

```
>
> # Extract scaled control store sales (already scaled)
> scaledSales <- scaledControlSales[YEARMONTH %in% trial_period,
+                                   .(YEARMONTH, controlSales)]
>
> # Merge and calculate percentage difference
> percentageDiff <- merge(trialSales,
+ print(percentageDiff)
+ percentageDiff <- merge(trialSales, scaledSales, by = "YEARMONTH")
Error: unexpected symbol in:
"print(percentageDiff)
percentageDiff"
> percentageDiff[, percentageDiff := (trialSales - controlSales) / controlSales]
Error: object 'percentageDiff' not found
> percentageDiff <- merge(trialSales, scaledSales, by = "YEARMONTH")
> percentageDiff[, percentageDiff := (trialSales - controlSales) / controlSales]
> print(percentageDiff)
Key: <YEARMONTH>
   YEARMONTH trialSales controlSales percentageDiff
       <num>      <num>        <num>          <num>
1:    201902      235.0     288.4955   -0.185429271
2:    201903      278.5     278.1657    0.001201622
3:    201904      263.5     284.4374   -0.073609842
> stdDev <- sd(percentageDiff[YEARMONTH < 201902 , percentageDiff])
> # Calculate standard deviation from pre-trial period differences
> stdDev <- sd(percentageDiff[YEARMONTH < 201902, percentageDiff])
>
> # Set degrees of freedom (8 months pre-trial => df = 8 - 1)
> degreesOfFreedom <- 7
>
> # Assume the null hypothesis mean difference (u) is 0
> # Calculate t-values for trial period
> percentageDiff[, tValue := (percentageDiff) / stdDev]
>
> #
> pastSales <- measureOverTimeSales[, Store_type :=
+                                   ][, totSales :=
Error: unexpected ']' in:
"pastSales <- measureOverTimeSales[, Store_type :=
                                 ]"
>                                   ][, TransactionMonth :=
Error: unexpected ']' in "                                    ]"
>                                   ][Store_type %in% c("Trial", "Control"), ]
Error: unexpected ']' in "                                    ]"
> # Set transaction month to Date format
> measureOverTimeSales[, TransactionMonth := as.Date(paste0(YEARMONTH, "01"), format = "%Y%m%d")]
>
> # Create labeled data for trial store
> trialData <- measureOverTimeSales[STORE_NBR == trial_store,
+                                   .(TransactionMonth, totSales, Store_type = "Trial Store")]
>
> # Create labeled data for scaled control store
> controlData <- scaledControlSales[, .(TransactionMonth, totSales = controlSales, Store_type = "
Control Store")]
>
> # Combine both into one dataset for plotting
> plotData <- rbind(trialData, controlData)
>
> # Plot total sales
> library(ggplot2)
>
> ggplot(plotData, aes(x = TransactionMonth, y = totSales, color = Store_type)) +
+   geom_line(size = 1.2) +
+   geom_vline(xintercept = as.Date("2019-02-01"), linetype = "dashed", color = "gray40") +
+   geom_vline(xintercept = as.Date("2019-04-30"), linetype = "dashed", color = "gray40") +
+   labs(title = "Total Sales: Trial vs Control Store",
+        subtitle = "Dashed lines indicate the trial period (Feb-Apr 2019)",
+        x = "Month", y = "Total Sales ($)", color = "Store Type") +
+   theme_minimal() +
+   theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
>  pastSales_Controls95 <- pastSales[Store_type == "Control",
+                                   ][, totSales := totSales * (1 + stdDev * 2)
+                                   ][, Store_type := "Control 95th % confidence
+ interval"]
>  pastSales_Controls5 <- pastSales[Store_type == "Control",
+                                   ][, totSales := totSales * (1 - stdDev * 2)
+                                   ][, Store_type := "Control 5th % confidence
+ interval"]
>  trialAssessment <- rbind(pastSales, pastSales_Controls95, pastSales_Controls5)
>  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 message:
Removed 16 rows containing missing values or values outside the scale range
(`geom_line()`).
> # Calculate the scaling factor for customer counts pre-trial
> scalingFactorForControlCustomers <- preTrialMeasures[STORE_NBR == trial_store & YEARMONTH < 201
902,
+                                                      sum(nCustomers)] /
+                                     preTrialMeasures[STORE_NBR == control_store & YEARMONTH < 2
01902,
+                                                      sum(nCustomers)]
> # Get full % difference data (not just trial period)
> customerDiffFull <- merge(trialCustomers, scaledControlCustomers, by = c("YEARMONTH", "Transact
ionMonth"))
Error: object 'trialCustomers' not found
> customerDiffFull[, percentageDiff := (trialCustomers - controlCustomers) / controlCustomers]
Error: object 'customerDiffFull' not found
>
> # Standard deviation of pre-trial % differences
> stdDevCust <- sd(customerDiffFull[YEARMONTH < 201902, percentageDiff])
Error: object 'customerDiffFull' not found
> degreesOfFreedom <- 7  # 8 months pre-trial
>
> # Compute t-values for trial period
> customerDiffFull[, tValue := percentageDiff / stdDevCust]
Error: object 'customerDiffFull' not found
>
> # 95th percentile of t-distribution
> tCriticalCust <- qt(0.95, df = degreesOfFreedom)
>
> # Extract trial months
> trialCustomerDiff <- customerDiffFull[YEARMONTH %in% 201902:201904]
Error: object 'customerDiffFull' not found
> stdDev <- sd(percentageDiff[YEARMONTH < 201902 , percentageDiff])
> degreesOfFreedom <- 7
>  pastCustomers <- measureOverTimeCusts[, nCusts := mean(nCustomers), by =
+ c("YEARMONTH", "Store_type")
+                                       ][Store_type %in% c("Trial", "Control"), ]
Error: object 'measureOverTimeCusts' not found
> measureOverTimeCusts <- measureOverTime
>  scaledControlCustomers <- measureOverTimeCusts[,
+                                                ][, controlCustomers :=
+                                                ][, Store_type :=
Error: unexpected ']' in:
"                                               ][, controlCustomers :=
                                               ]"
>                                     ]
Error: unexpected ']' in "                                             ]"
> scalingFactorForControlCust <-
+ measureOverTimeCusts <- measureOverTime
>  scaledControlCustomers <- measureOverTimeCusts[,
+                                                ][, controlCustomers :=
+                                                ][, Store_type :=
Error: unexpected ']' in:
"                                               ][, controlCustomers :=
                                               ]"
```

```
>                                  ]
Error: unexpected ']' in "                                  ]"
>  percentageDiff <-
+ )
Error: unexpected ')' in:
" percentageDiff <-
)"
> # 1. Compute the scaling factor
> scalingFactorForControlCust <- preTrialMeasures[STORE_NBR == trial_store & YEARMONTH < 201902,
sum(nCustomers)] /
+                                preTrialMeasures[STORE_NBR == control_store & YEARMONTH < 201902
, sum(nCustomers)]
>
> # 2. Apply the scaling factor to control store
> measureOverTimeCusts <- measureOverTime
>
> scaledControlCustomers <- measureOverTimeCusts[STORE_NBR == control_store,
+                                .(YEARMONTH,
+                                  controlCustomers = nCustomers * scalingFactorForControlCust,
+                                  Store_type = "Control")]
>
> # 3. Extract trial store customer data
> trialCustomers <- measureOverTimeCusts[STORE_NBR == trial_store,
+                                .(YEARMONTH,
+                                  trialCustomers = nCustomers,
+                                  Store_type = "Trial")]
>
> # 4. Merge and calculate percentage difference
> percentageDiff <- merge(trialCustomers, scaledControlCustomers, by = "YEARMONTH")[,
+ percentageDiff := (trialCustomers - controlCustomers) / controlCustomers]
> # Calculate t-values
> percentageDiff[, tValue := percentageDiff / stdDev]
>
> # Add TransactionMonth for plotting
> percentageDiff[, TransactionMonth := as.Date(paste0(YEARMONTH, "01"), format = "%Y%m%d")]
>
> # Filter only trial period months
> trialMonths <- percentageDiff[YEARMONTH %in% 201902:201904]
>
> # Get critical t-value at 95% confidence
> tCritical <- qt(0.95, df = degreesOfFreedom)
>
> # Plot the t-values with significance line
> library(ggplot2)
>
> ggplot(trialMonths, aes(x = TransactionMonth, y = tValue)) +
+   geom_col(fill = "steelblue") +
+   geom_hline(yintercept = tCritical, linetype = "dashed", color = "red", size = 1) +
+   labs(
+     title = "Customer Count: T-Values for Trial vs Scaled Control",
+     subtitle = "Red dashed line = 95th percentile of t-distribution (significance threshold)",
+     x = "Month",
+     y = "T-Value"
+   ) +
+   theme_minimal()
Error in seq.int(0, to0 - from, by) : 'to' must be a finite number
> pastCustomers <- measureOverTimeCusts[, nCusts := mean(nCustomers), by =
+ c("YEARMONTH", "Store_type")
+                                ][Store_type %in% c("Trial", "Control"), ]
>  pastCustomers_Controls95 <- pastCustomers[Store_type == "Control",
+                                ][, nCusts := nCusts * (1 + stdDev * 2)
+                                ][, Store_type := "Control 95th % confidence
+ interval"]
>  pastCustomers_Controls5 <- pastCustomers[Store_type == "Control",
+                                ][, nCusts := nCusts * (1 - stdDev * 2)
+                                ][, Store_type := "Control 5th % confidence
+ interval"]
>  trialAssessment <- rbind(pastCustomers, pastCustomers_Controls95,
+ pastCustomers_Controls5)
>  ggplot() +
```

```
+   geom_rect(data = , aes(xmin = , xmax = ,  ymin =  , ymax = , color = ),
+ show.legend = FALSE) +
+   geom_line() +
+   labs()
>  measureOverTime <- data[, .(totSales = ,
+                               nCustomers = ,
+                               nTxnPerCust = ,
+                               nChipsPerTxn = ,
+                               avgPricePerUnit =
+                               )
+                           , by = ][order(, )]
Error in list(, , , , ) : argument 1 is empty
> library(data.table)
>
> # Assuming `data` is your loaded dataset
> measureOverTime <- data[, .(
+   totSales = sum(TOT_SALES),
+   nCustomers = uniqueN(LYLTY_CARD_NBR),
+   nTxnPerCust = .N / uniqueN(LYLTY_CARD_NBR),
+   nChipsPerTxn = sum(PROD_QTY) / .N,
+   avgPricePerUnit = sum(TOT_SALES) / sum(PROD_QTY)
+ ), by = .(STORE_NBR, YEARMONTH)][order(STORE_NBR, YEARMONTH)]
> # Set the trial store
> trial_store <- 86
>
> # Calculate correlation for total sales
> corr_nSales <- calculateCorrelation(data = measureOverTime,
+                                     trial_store = trial_store,
+                                     measure = "totSales")
Error in calculateCorrelation(data = measureOverTime, trial_store = trial_store,  :
  unused arguments (data = measureOverTime, trial_store = trial_store, measure = "totSales")
>
> # Calculate correlation for number of customers
> corr_nCustomers <- calculateCorrelation(data = measureOverTime,
+                                         trial_store = trial_store,
+                                         measure = "nCustomers")
Error in calculateCorrelation(data = measureOverTime, trial_store = trial_store,  :
  unused arguments (data = measureOverTime, trial_store = trial_store, measure = "nCustomers")
>
> # Calculate magnitude similarity for total sales
> magnitude_nSales <- calculateMagnitudeDistance(data = measureOverTime,
+                                                trial_store = trial_store,
+                                                measure = "totSales")
Error in calculateMagnitudeDistance(data = measureOverTime, trial_store = trial_store,  :
  unused arguments (data = measureOverTime, trial_store = trial_store, measure = "totSales")
>
> # Calculate magnitude similarity for number of customers
> magnitude_nCustomers <- calculateMagnitudeDistance(data = measureOverTime,
+                                                    trial_store = trial_store,
+                                                    measure = "nCustomers")
Error in calculateMagnitudeDistance(data = measureOverTime, trial_store = trial_store,  :
  unused arguments (data = measureOverTime, trial_store = trial_store, measure = "nCustomers")
> # Set the trial store
> trial_store <- 86
>
> # Calculate correlation for total sales
> corr_nSales <- calculateCorrelation(data = measureOverTime,
+                                     trial_store = trial_store,
+                                     measure = "totSales")
Error in calculateCorrelation(data = measureOverTime, trial_store = trial_store,  :
  unused arguments (data = measureOverTime, trial_store = trial_store, measure = "totSales")
>
> # Calculate correlation for number of customers
> corr_nCustomers <- calculateCorrelation(data = measureOverTime,
+                                         trial_store = trial_store,
+                                         measure = "nCustomers")
Error in calculateCorrelation(data = measureOverTime, trial_store = trial_store,  :
  unused arguments (data = measureOverTime, trial_store = trial_store, measure = "nCustomers")
>
> # Calculate magnitude similarity for total sales
```

```
> magnitude_nSales <- calculateMagnitudeDistance(data = measureOverTime,
+                                                trial_store = trial_store,
+                                                measure = "totSales")
Error in calculateMagnitudeDistance(data = measureOverTime, trial_store = trial_store,  :
  unused arguments (data = measureOverTime, trial_store = trial_store, measure = "totSales")
>
> # Calculate magnitude similarity for number of customers
> magnitude_nCustomers <- calculateMagnitudeDistance(data = measureOverTime,
+                                                    trial_store = trial_store,
+                                                    measure = "nCustomers")
Error in calculateMagnitudeDistance(data = measureOverTime, trial_store = trial_store,  :
  unused arguments (data = measureOverTime, trial_store = trial_store, measure = "nCustomers")
> # Set the trial store
> trial_store <- 86
>
> # Calculate correlation for total sales
> corr_nSales <- calculateCorrelation(data = measureOverTime,
+                                     trial_store = trial_store,
+                                     measure = "totSales")
Error in calculateCorrelation(data = measureOverTime, trial_store = trial_store,  :
  unused arguments (data = measureOverTime, trial_store = trial_store, measure = "totSales")
>
> # Calculate correlation for number of customers
> corr_nCustomers <- calculateCorrelation(data = measureOverTime,
+                                         trial_store = trial_store,
+                                         measure = "nCustomers")
Error in calculateCorrelation(data = measureOverTime, trial_store = trial_store,  :
  unused arguments (data = measureOverTime, trial_store = trial_store, measure = "nCustomers")
>
> # Calculate magnitude similarity for total sales
> magnitude_nSales <- calculateMagnitudeDistance(data = measureOverTime,
+                                                trial_store = trial_store,
+                                                measure = "totSales")
Error in calculateMagnitudeDistance(data = measureOverTime, trial_store = trial_store,  :
  unused arguments (data = measureOverTime, trial_store = trial_store, measure = "totSales")
>
> # Calculate magnitude similarity for number of customers
> magnitude_nCustomers <- calculateMagnitudeDistance(data = measureOverTime,
+                                                    trial_store = trial_store,
+                                                    measure = "nCustomers")
Error in calculateMagnitudeDistance(data = measureOverTime, trial_store = trial_store,  :
  unused arguments (data = measureOverTime, trial_store = trial_store, measure = "nCustomers")
> library(data.table)
>
> # Assuming `data` is your loaded dataset
> measureOverTime <- data[, .(
+   totSales = sum(TOT_SALES),
+   nCustomers = uniqueN(LYLTY_CARD_NBR),
+   nTxnPerCust = .N / uniqueN(LYLTY_CARD_NBR),
+   nChipsPerTxn = sum(PROD_QTY) / .N,
+   avgPricePerUnit = sum(TOT_SALES) / sum(PROD_QTY)
+ ), by = .(STORE_NBR, YEARMONTH)][order(STORE_NBR, YEARMONTH)]
> ...
Error: '...' used in an incorrect context
> (control_store output above remains unchanged)
Error: unexpected symbol in "(control_store output"
>
> ```{r Visual check - Total Sales}
Error: attempt to use zero-length variable name
> measureOverTimeSales <- measureOverTime
> pastSales <- measureOverTimeSales[, Store_type := ifelse(STORE_NBR == trial_store, "Trial",
+                                                          ifelse(STORE_NBR == control_store, "Con
trol", "Other stores"))
+                                   ][, totSales := mean(totSales), by = c("YEARMONTH", "Store_type")
+                                   ][, TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH
 %% 100, 1, sep = "-"), "%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")
> ```
Error: attempt to use zero-length variable name
>
> ```{r Visual check - Number of Customers}
Error: attempt to use zero-length variable name
> measureOverTimeCusts <- measureOverTime
> 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_typ
e")
+   ][, TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH
 %% 100, 1, sep = "-"), "%Y-%m-%d")
+   ][YEARMONTH < 201903 ,]
Warning message:
In `[.data.table`(measureOverTimeCusts[, `:=`(Store_type, ifelse(STORE_NBR == :
  70.575758 (type 'double') at RHS position 1 out-of-range(NA) or truncated (precision lost) when
 assigning to type 'integer' (column 4 named 'nCustomers')
>
> ggplot(pastCustomers, aes(TransactionMonth, nCustomers, color = Store_type)) +
+   geom_line() +
+   labs(x = "Month of operation", y = "Number of customers", title = "Total customers by month")
> ```
Error: attempt to use zero-length variable name
>
> ```{r Trial impact - Total Sales}
Error: attempt to use zero-length variable name
> scalingFactorForControlSales <- preTrialMeasures[STORE_NBR == trial_store, sum(totSales)]/
+   preTrialMeasures[STORE_NBR == control_store, sum(totSales)]
>
> scaledControlSales <- measureOverTime[STORE_NBR == control_store, .(YEARMONTH, totSales)]
> scaledControlSales[, controlSales := totSales * scalingFactorForControlSales]
>
> trialSales <- measureOverTime[STORE_NBR == trial_store, .(YEARMONTH, trialSales = totSales)]
> percentageDiff <- merge(scaledControlSales[, .(YEARMONTH, controlSales)], trialSales, by = "YEA
RMONTH")
> percentageDiff[, percentageDiff := abs(trialSales - controlSales)/controlSales]
>
> stdDev <- sd(percentageDiff[YEARMONTH < 201902, percentageDiff])
> degreesOfFreedom <- 7
>
> percentageDiff[, tValue := (percentageDiff - 0)/stdDev
+   ][, TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH %% 100, 1, se
p = "-"), "%Y-%m-%d")
+   ][YEARMONTH %in% 201902:201904, .(YEARMONTH, percentageDiff, tValue)]
Key: <YEARMONTH>
   YEARMONTH percentageDiff   tValue
       <num>   <num>   <num>
1:    201902   0.12122146 1.0613908
2:    201903   0.02478983 0.2170548
3:    201904   0.17212612 1.5071018
> ```
Error: attempt to use zero-length variable name
>
> ```{r Visualize trial impact - Total Sales}
Error: attempt to use zero-length variable name
> pastSales <- measureOverTime[, Store_type := ifelse(STORE_NBR == trial_store, "Trial",
+   ifelse(STORE_NBR == control_store, "Control
", NA))
+   ][!is.na(Store_type), totSales := sum(totSales), by = .(YEARMONTH,
 Store_type)
+   ][, TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH
 %% 100, 1, sep = "-"), "%Y-%m-%d")
+   ][YEARMONTH >= 201807 & YEARMONTH <= 201904]
>
> pastSales_Controls95 <- pastSales[Store_type == "Control"][, totSales := totSales * (1 + stdDev
 * 2)][, Store_type := "Control 95th % confidence interval"]
> pastSales_Controls5 <- pastSales[Store_type == "Control"][, totSales := totSales * (1 - stdDev
* 2)][, Store_type := "Control 5th % confidence interval"]
```

```
>
> trialAssessment <- rbind(pastSales, pastSales_Controls95, pastSales_Controls5)
>
> ggplot(trialAssessment, aes(TransactionMonth, totSales, color = Store_type)) +
+   geom_rect(data = trialAssessment[YEARMONTH >= 201902 & YEARMONTH <= 201904],
+             aes(xmin = min(TransactionMonth), xmax = max(TransactionMonth), ymin = 0, ymax = In
f),
+             fill = "grey", alpha = 0.2, inherit.aes = FALSE, show.legend = FALSE) +
+   geom_line() +
+   labs(x = "Month of operation", y = "Total sales", title = "Trial vs Control Store Sales")
Warning message:
In geom_rect(data = trialAssessment[YEARMONTH >= 201902 & YEARMONTH <=  :
  All aesthetics have length 1, but the data has 800 rows.
ⓘ Please consider using `annotate()` or provide this layer with data containing a single
  row.
> ```
Error: attempt to use zero-length variable name
>
> ```{r Trial impact - Number of Customers}
Error: attempt to use zero-length variable name
> scalingFactorForControlCust <- preTrialMeasures[STORE_NBR == trial_store, sum(nCustomers)]/
+   preTrialMeasures[STORE_NBR == control_store, sum(nCustomers)]
>
> scaledControlCustomers <- measureOverTime[STORE_NBR == control_store, .(YEARMONTH, nCustomers)]
> scaledControlCustomers[, controlCustomers := nCustomers * scalingFactorForControlCust]
>
> trialCustomers <- measureOverTime[STORE_NBR == trial_store, .(YEARMONTH, trialCustomers = nCust
omers)]
> percentageDiffCust <- merge(scaledControlCustomers[, .(YEARMONTH, controlCustomers)], trialCust
omers, by = "YEARMONTH")
> percentageDiffCust[, percentageDiff := abs(trialCustomers - controlCustomers)/controlCustomers]
>
> stdDevCust <- sd(percentageDiffCust[YEARMONTH < 201902, percentageDiff])
>
> pastCustomers <- measureOverTime[, Store_type := ifelse(STORE_NBR == trial_store, "Trial",
+   ifelse(STORE_NBR == control_store, "Con
trol", NA))
+   ][!is.na(Store_type), nCustomers := sum(nCustomers), by = .(YE
ARMONTH, Store_type)
+   ][, TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARM
ONTH %% 100, 1, sep = "-"), "%Y-%m-%d")
+   ][YEARMONTH >= 201807 & YEARMONTH <= 201904]
>
> pastCustomers_Controls95 <- pastCustomers[Store_type == "Control"][, nCustomers := nCustomers *
 (1 + stdDevCust * 2)][, Store_type := "Control 95th % confidence interval"]
> pastCustomers_Controls5 <- pastCustomers[Store_type == "Control"][, nCustomers := nCustomers *
(1 - stdDevCust * 2)][, Store_type := "Control 5th % confidence interval"]
>
> trialAssessmentCust <- rbind(pastCustomers, pastCustomers_Controls95, pastCustomers_Controls5)
>
> ggplot(trialAssessmentCust, aes(TransactionMonth, nCustomers, color = Store_type)) +
+   geom_rect(data = trialAssessmentCust[YEARMONTH >= 201902 & YEARMONTH <= 201904],
+   aes(xmin = min(TransactionMonth), xmax = max(TransactionMonth), ymin = 0, ymax = In
f),
+   fill = "grey", alpha = 0.2, inherit.aes = FALSE, show.legend = FALSE) +
+   geom_line() +
+   labs(x = "Month of operation", y = "Number of customers", title = "Trial vs Control Store Cus
tomers")
Warning message:
In geom_rect(data = trialAssessmentCust[YEARMONTH >= 201902 & YEARMONTH <= :
  All aesthetics have length 1, but the data has 800 rows.
ⓘ Please consider using `annotate()` or provide this layer with data containing a single
  row.
> ```
Error: attempt to use zero-length variable name
>
> # Analysis complete for trial store 77. Repeat for trial stores 86 and 88...
> ...(existing code for trial store 77 remains unchanged)
Error: unexpected symbol in "...(existing code"
>
```

```
> # Repeat for trial store 86
> ```{r Analysis - Trial Store 86}
Error: attempt to use zero-length variable name
> trial_store <- 86
>
> corr_nSales <- calculateCorrelation(preTrialMeasures, quote(totSales), trial_store)
Error in calculateCorrelation(preTrialMeasures, quote(totSales), trial_store) :
  object 'storeNumbers' not found
> corr_nCustomers <- calculateCorrelation(preTrialMeasures, quote(nCustomers), trial_store)
Error in calculateCorrelation(preTrialMeasures, quote(nCustomers), trial_store) :
  object 'storeNumbers' not found
>
> magnitude_nSales <- calculateMagnitudeDistance(preTrialMeasures, quote(totSales), trial_store)
> magnitude_nCustomers <- calculateMagnitudeDistance(preTrialMeasures, quote(nCustomers), trial_s
tore)
>
> score_nSales <- merge(corr_nSales, magnitude_nSales, by = c("Store1", "Store2"))[, scoreNSales
:= corr_measure * corr_weight + mag_measure * (1 - corr_weight)]
Error: object 'corr_nSales' not found
> score_nCustomers <- merge(corr_nCustomers, magnitude_nCustomers, by = c("Store1", "Store2"))[,
scoreNCust := corr_measure * corr_weight + mag_measure * (1 - corr_weight)]
Error: object 'corr_nCustomers' not found
>
> score_Control <- merge(score_nSales[, .(Store1, Store2, scoreNSales)], score_nCustomers[, .(Sto
re1, Store2, scoreNCust)], by = c("Store1", "Store2"))
Error in eval(jsub, SDenv, parent.frame()) : object 'Store1' not found
> score_Control[, finalControlScore := scoreNSales * 0.5 + scoreNCust * 0.5]
>
> control_store <- score_Control[Store1 == trial_store][order(-finalControlScore)][1, Store2]
Error in .checkTypos(e, names_x) :
  Object 'Store1' not found. Perhaps you intended [Control_Store]
> control_store
[1] 41
> ```
Error: attempt to use zero-length variable name
>
> # Repeat for trial store 88
> ```{r Analysis - Trial Store 88}
Error: attempt to use zero-length variable name
> trial_store <- 88
>
> corr_nSales <- calculateCorrelation(preTrialMeasures, quote(totSales), trial_store)
Error in calculateCorrelation(preTrialMeasures, quote(totSales), trial_store) :
  object 'storeNumbers' not found
> corr_nCustomers <- calculateCorrelation(preTrialMeasures, quote(nCustomers), trial_store)
Error in calculateCorrelation(preTrialMeasures, quote(nCustomers), trial_store) :
  object 'storeNumbers' not found
>
> magnitude_nSales <- calculateMagnitudeDistance(preTrialMeasures, quote(totSales), trial_store)
> magnitude_nCustomers <- calculateMagnitudeDistance(preTrialMeasures, quote(nCustomers), trial_s
tore)
>
> score_nSales <- merge(corr_nSales, magnitude_nSales, by = c("Store1", "Store2"))[, scoreNSales
:= corr_measure * corr_weight + mag_measure * (1 - corr_weight)]
Error: object 'corr_nSales' not found
> score_nCustomers <- merge(corr_nCustomers, magnitude_nCustomers, by = c("Store1", "Store2"))[,
scoreNCust := corr_measure * corr_weight + mag_measure * (1 - corr_weight)]
Error: object 'corr_nCustomers' not found
>
> score_Control <- merge(score_nSales[, .(Store1, Store2, scoreNSales)], score_nCustomers[, .(Sto
re1, Store2, scoreNCust)], by = c("Store1", "Store2"))
Error in eval(jsub, SDenv, parent.frame()) : object 'Store1' not found
> score_Control[, finalControlScore := scoreNSales * 0.5 + scoreNCust * 0.5]
>
> control_store <- score_Control[Store1 == trial_store][order(-finalControlScore)][1, Store2]
Error in .checkTypos(e, names_x) :
  Object 'Store1' not found. Perhaps you intended [Control_Store]
> control_store
[1] 41
> ```
```

```
Error: attempt to use zero-length variable name
>
> # Visual and statistical assessment for store 86 and 88 would follow the same structure as for
store 77, using the same functions and plotting logic. You can reuse and wrap them into functions
 for conciseness.
> ...
Error: '...' used in an incorrect context
>
> # Reusable functions for assessment
> ```{r Define reusable functions for plotting and significance}
Error: attempt to use zero-length variable name
> assess_trial <- function(trial_store, control_store, measureOverTime, valueCol) {
+   # Scale control store to trial store
+   scalingFactor <- preTrialMeasures[STORE_NBR == trial_store, sum(eval(valueCol))] /
+     preTrialMeasures[STORE_NBR == control_store, sum(eval(valueCol))]
+
+   trialMeasure <- measureOverTime[STORE_NBR == trial_store, .(YEARMONTH, trialVal = eval(valueC
ol))]
+   controlMeasure <- measureOverTime[STORE_NBR == control_store, .(YEARMONTH, controlVal = eval(
valueCol))]
+   controlMeasure[, controlVal := controlVal * scalingFactor]
+
+   percentageDiff <- merge(controlMeasure, trialMeasure, by = "YEARMONTH")
+   percentageDiff[, percentageDiff := abs(trialVal - controlVal)/controlVal]
+
+   stdDev <- sd(percentageDiff[YEARMONTH < 201902, percentageDiff])
+   percentageDiff[, tValue := (percentageDiff - 0) / stdDev]
+   percentageDiff[, TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH %% 100, 1, se
p = "-"), "%Y-%m-%d")]
+
+   # Plotting
+   measureType <- deparse(substitute(valueCol))
+   measureSummary <- measureOverTime[, Store_type := ifelse(STORE_NBR == trial_store, "Trial",
+                                                            ifelse(STORE_NBR == control_store, "
Control", NA))
+                                     ][!is.na(Store_type), value := eval(valueCol),
+                                       by = .(YEARMONTH, Store_type)]
+   measureSummary[, TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH %% 100, 1, se
p = "-"), "%Y-%m-%d")]
+
+   control95 <- measureSummary[Store_type == "Control"]
+   control95[, value := value * (1 + stdDev * 2)]
+   control95[, Store_type := "Control 95th % confidence interval"]
+
+   control5 <- measureSummary[Store_type == "Control"]
+   control5[, value := value * (1 - stdDev * 2)]
+   control5[, Store_type := "Control 5th % confidence interval"]
+
+   combined <- rbind(measureSummary, control95, control5)
+
+   p <- ggplot(combined, aes(TransactionMonth, value, color = Store_type)) +
+     geom_rect(data = combined[YEARMONTH >= 201902 & YEARMONTH <= 201904],
+               aes(xmin = min(TransactionMonth), xmax = max(TransactionMonth), ymin = 0, ymax =
Inf),
+               fill = "grey", alpha = 0.2, inherit.aes = FALSE, show.legend = FALSE) +
+     geom_line() +
+     labs(x = "Month of operation",
+          y = measureType,
+          title = paste("Trial vs Control Store:", measureType))
+   print(p)
+   return(percentageDiff[YEARMONTH >= 201902 & YEARMONTH <= 201904, .(YEARMONTH, percentageDiff,
 tValue)])
+ }
> ```
Error: attempt to use zero-length variable name
>
> # Apply reusable function to trial stores
> ```{r Assess trial store 86}
Error: attempt to use zero-length variable name
> assess_trial(trial_store = 86, control_store = control_store, measureOverTime, quote(totSales))
```

```
Key: <YEARMONTH>
   YEARMONTH percentageDiff    tValue
       <num>          <num>     <num>
1:    201902     0.12122146 1.0613908
2:    201903     0.02478983 0.2170548
3:    201904     0.17212612 1.5071018
Warning messages:
1: In geom_rect(data = combined[YEARMONTH >= 201902 & YEARMONTH <=  :
  All aesthetics have length 1, but the data has 800 rows.
ⓘ Please consider using `annotate()` or provide this layer with data containing a single
  row.
2: Removed 3145 rows containing missing values or values outside the scale range
(`geom_line()`).
> assess_trial(trial_store = 86, control_store = control_store, measureOverTime, quote(nCustomers
))
Key: <YEARMONTH>
   YEARMONTH percentageDiff    tValue
       <num>          <num>     <num>
1:    201902     0.08505022 0.7201973
2:    201903     0.06886657 0.5831557
3:    201904     0.04484264 0.3797233
Warning messages:
1: In geom_rect(data = combined[YEARMONTH >= 201902 & YEARMONTH <=  :
  All aesthetics have length 1, but the data has 800 rows.
ⓘ Please consider using `annotate()` or provide this layer with data containing a single
  row.
2: Removed 3145 rows containing missing values or values outside the scale range
(`geom_line()`).
> ```
Error: attempt to use zero-length variable name
>
> ```{r Assess trial store 88}
Error: attempt to use zero-length variable name
> assess_trial(trial_store = 88, control_store = control_store, measureOverTime, quote(totSales))
Key: <YEARMONTH>
   YEARMONTH percentageDiff   tValue
       <num>   <num>   <num>
1:    201902   0.6418420 9.604692
2:    201903   0.5916049 8.852931
3:    201904   0.6157288 9.213928
Warning messages:
1: In geom_rect(data = combined[YEARMONTH >= 201902 & YEARMONTH <= :
  All aesthetics have length 1, but the data has 800 rows.
ⓘ Please consider using `annotate()` or provide this layer with data containing a single
  row.
2: Removed 262 rows containing missing values or values outside the scale range
(`geom_line()`).
> assess_trial(trial_store = 88, control_store = control_store, measureOverTime, quote(nCustomers
))
Key: <YEARMONTH>
   YEARMONTH percentageDiff   tValue
       <num>   <num>   <num>
1:    201902   0.5597727 6.938615
2:    201903   0.4773221 5.916606
3:    201904   0.5100580 6.322381
Warning messages:
1: In geom_rect(data = combined[YEARMONTH >= 201902 & YEARMONTH <= :
  All aesthetics have length 1, but the data has 800 rows.
ⓘ Please consider using `annotate()` or provide this layer with data containing a single
  row.
2: Removed 262 rows containing missing values or values outside the scale range
(`geom_line()`).
>
```