Quantium Virtual Internship - Retail Strategy and Analytics - Task 2

Yussuf Ali

2024-09-15

Load required libraries and datasets

Note that you will need to install these libraries if you have never used these before.

install.packages("tidyr")

```
getwd()
```

```
## [1] "C:/Users/USER/Desktop/Python program/Data_Analytics_internship"
```

```
setwd("C:/Users/USER/Desktop/Python program/Data_Analytics_internship/")


filePath <- "C:/Users/USER/Desktop/Python program/Data_Analytics_internship/"
```

```
data <- fread(paste0(filePath,"QVI_data.csv"))
#### Set themes for plots
theme_set(theme_bw())
theme_update(plot.title = element_text(hjust = 0.5))
```

```
# Create a new column YEARMONTH in the format yyyymm
data[, YEARMONTH := format(as.Date(DATE), "%Y%m")]
#head(data)

# Calculate total sales, number of customers, transactions per customer, chips
 ↪  per transaction, and average price per unit
measureOverTime <- data[, .(
  totSales = sum(TOT_SALES),                                  # Total sales per store
   ↪  per month
  nCustomers = uniqueN(LYLTY_CARD_NBR),                        # Unique customers per
   ↪  store per month
  nTxnPerCust = .N / uniqueN(LYLTY_CARD_NBR),                  # Transactions per
   ↪  customer
  nChipsPerTxn = sum(PROD_QTY) / .N,                          # Chips per transaction
  avgPricePerUnit = sum(TOT_SALES) / sum(PROD_QTY)             # Average price per
   ↪  chip unit
), by = .(STORE_NBR, YEARMONTH)][order(STORE_NBR, YEARMONTH)]
```

```r
# View the first few rows of the calculated measures
head(measureOverTime)
```

```
##    STORE_NBR YEARMONTH totSales nCustomers nTxnPerCust nChipsPerTxn
##        <int>    <char>    <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
```

```r
# Identify stores with full observation periods (12 months)
storesWithFullObs <- unique(measureOverTime[, .N, by = STORE_NBR][N == 12,
 ↪  STORE_NBR])

# Filter to the pre-trial period (before February 2019) for stores with full
 ↪  observation periods
preTrialMeasures <- measureOverTime[YEARMONTH < 201902 & STORE_NBR %in%
 ↪  storesWithFullObs, ]

# View
head(preTrialMeasures)
```

```
##    STORE_NBR YEARMONTH totSales nCustomers nTxnPerCust nChipsPerTxn
##        <int>    <char>    <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
```

```
calculateCorrelation <- function(inputTable, metricCol, storeComparison) {
  calcCorrTable <- data.table(Store1 = numeric(), Store2 = numeric(),
  ↪  corr_measure = numeric())

  storeNumbers <- unique(inputTable[, STORE_NBR])

  for (i in storeNumbers) {
    calculatedMeasure <- data.table(
      "Store1" = storeComparison,
      "Store2" = i,
      "corr_measure" = cor(
        inputTable[STORE_NBR == storeComparison, eval(metricCol)],
        inputTable[STORE_NBR == i, eval(metricCol)]
      )
    )

    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)
  }

  # Standardize 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)
}
```

```
trial_store <- 77

corr_nSales <- calculateCorrelation(preTrialMeasures, quote(totSales),
 ↪  trial_store)
corr_nCustomers <- calculateCorrelation(preTrialMeasures, quote(nCustomers),
 ↪  trial_store)

# Use the functions for calculating magnitude
magnitude_nSales <- calculateMagnitudeDistance(preTrialMeasures,
 ↪  quote(totSales), trial_store)
magnitude_nCustomers <- calculateMagnitudeDistance(preTrialMeasures,
 ↪  quote(nCustomers), trial_store)
```

```
corr_weight <- 0.5

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

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

```
# Combine the scores for total sales and number of customers
score_Control <- merge(score_nSales, score_nCustomers, by = c("Store1",
 ↪  "Store2"))

# Calculate the final control score as a simple average of the sales and
 ↪  customer scores
score_Control[, finalControlScore := scoreNSales * 0.5 + scoreNCust * 0.5]

head(score_Control)
```

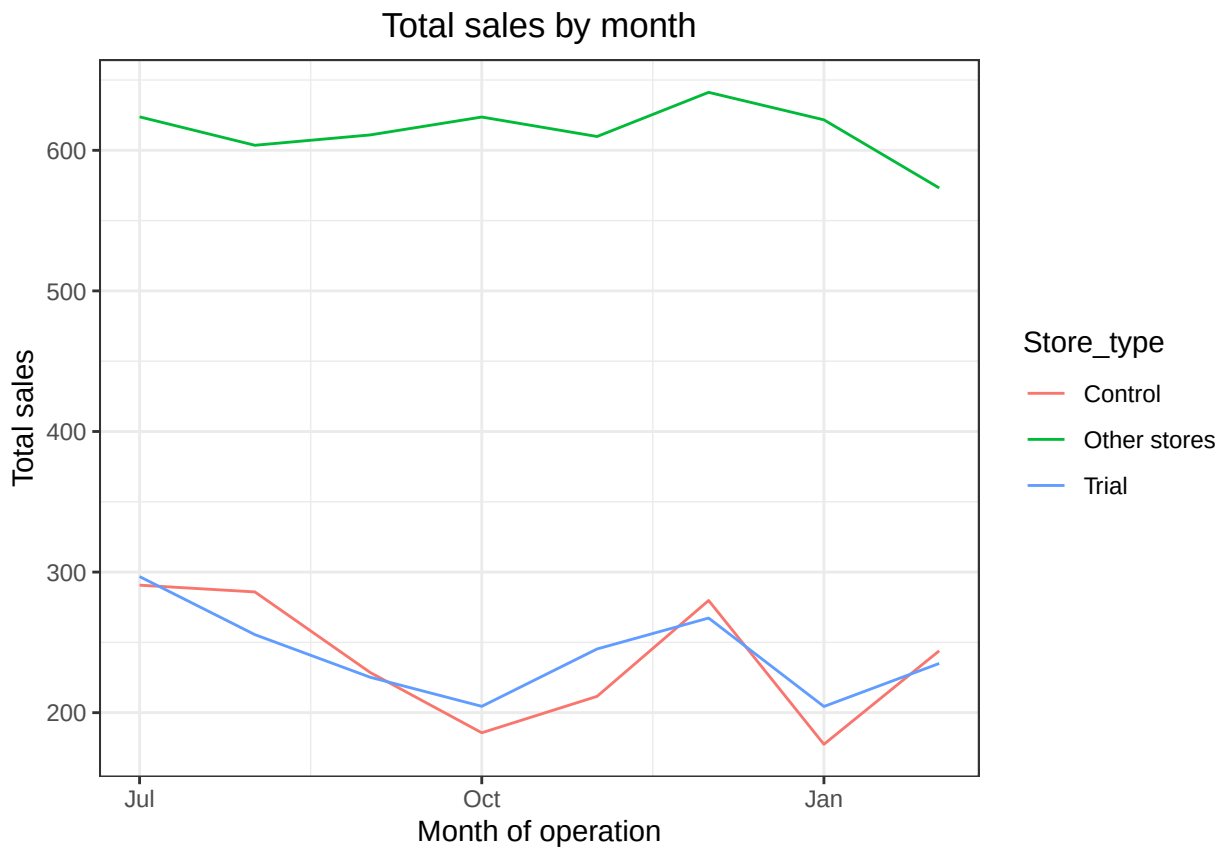
```
## Key: <Store1, Store2>
##    Store1 Store2 corr_measure.x mag_measure.x scoreNSales corr_measure.y
##     <num>  <num>          <num>         <num>       <num>          <num>
## 1:     77      1     0.07521784     0.9532849  0.51425135      0.3221683
## 2:     77      2    -0.26307873     0.9375792  0.33725024     -0.5720509
## 3:     77      3     0.80664364     0.3543149  0.58047929      0.8342074
## 4:     77      4    -0.26329960     0.1771353 -0.04308215     -0.2956387
## 5:     77      5    -0.11065231     0.5530434  0.22119557      0.3706585
## 6:     77      6     0.04248975     0.9692924  0.50589107      0.1368555
##    mag_measure.y  scoreNCust finalControlScore
##            <num>       <num>             <num>
## 1:     0.9403206  0.63124446        0.57274791
## 2:     0.9246380  0.17629355        0.25677189
## 3:     0.3450667  0.58963705        0.58505817
## 4:     0.1895787 -0.05303001       -0.04805608
```

```
## 5:     0.4811990  0.42592875        0.32356216
## 6:     0.9396196  0.53823759        0.52206433
```

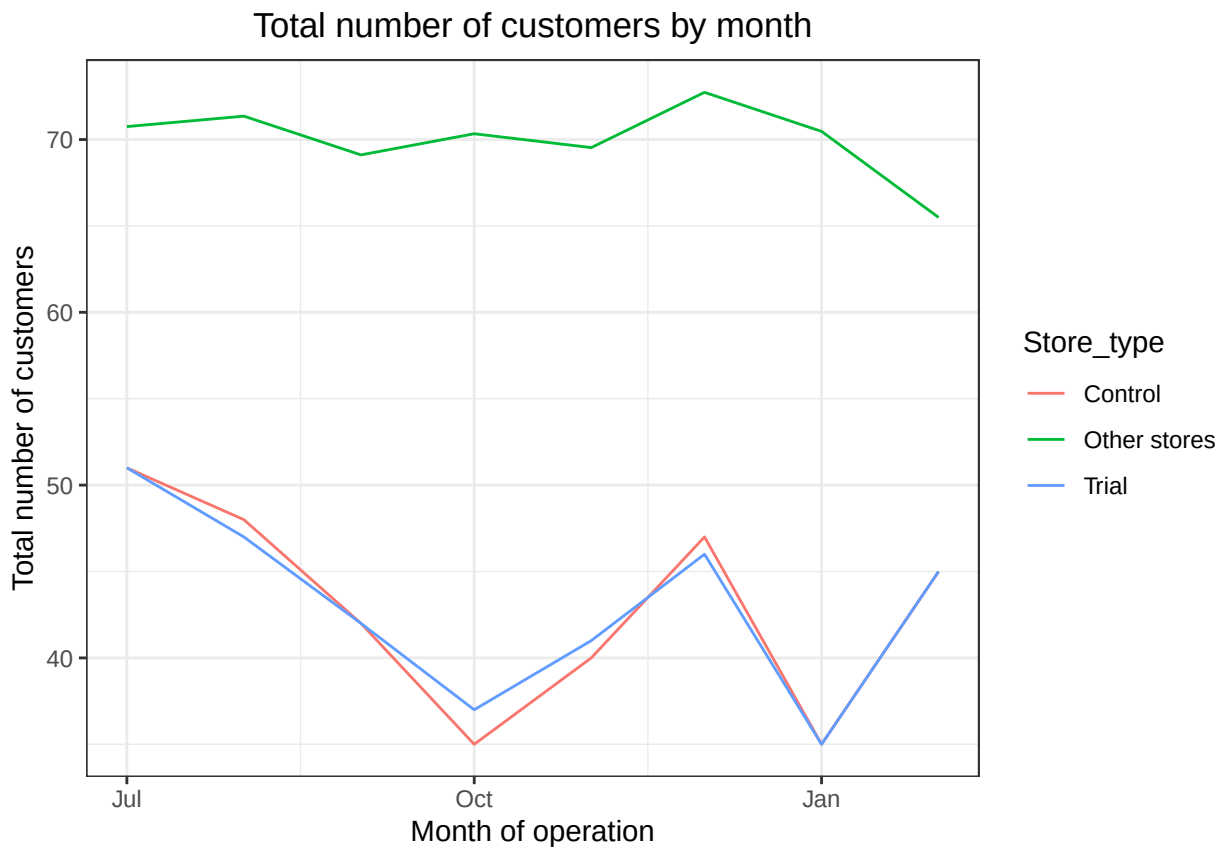
```
# Select the control store based on the highest final control score (excluding
 ↪  the trial store itself)
control_store <- score_Control[Store1 ==
 ↪  trial_store][order(-finalControlScore)][2, Store2]

# Display the selected control store for trial store 77
control_store
```

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

```
# 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(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")
```




```
# Visual checks on trends for number of customers
measureOverTimeCusts <- measureOverTime

pastCustomers <- measureOverTimeCusts[
  , Store_type := ifelse(STORE_NBR == trial_store, "Trial",
                         ifelse(STORE_NBR == control_store, "Control", "Other
                          ↪  stores"))
][
  , numberCustomers := mean(nCustomers), by = c("YEARMONTH", "Store_type")
][
  , TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH %% 100, 1,
 ↪  sep = "-"), "%Y-%m-%d")
][
  YEARMONTH < 201903,
]

ggplot(pastCustomers, aes(TransactionMonth, numberCustomers, color =
 ↪  Store_type)) +
  geom_line() +
  labs(x = "Month of operation", y = "Total number of customers", title = "Total
   ↪  number of customers by month")
```




```
# Calculate scaling factor for control 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
]
```

```
# Calculate percentage difference between scaled control sales and trial store
 ↪  sales
percentageDiff <- merge(
  scaledControlSales[, .(YEARMONTH, controlSales)],
  measureOverTime[STORE_NBR == trial_store, .(totSales, YEARMONTH)],
  by = "YEARMONTH"
)[
  , percentageDiff := abs(controlSales - totSales) / controlSales
]
```

```r
# View the result
head(percentageDiff)
```

```
## Key: <YEARMONTH>
##    YEARMONTH controlSales totSales percentageDiff
##        <num>        <num>    <num>          <num>
## 1:    201807     297.5656    296.8    0.002572711
## 2:    201808     292.6522    255.5    0.126949972
## 3:    201809     233.9989    225.2    0.037602377
## 4:    201810     190.0857    204.5    0.075830345
## 5:    201811     216.5974    245.3    0.132515791
## 6:    201812     286.4081    267.3    0.066716409
```

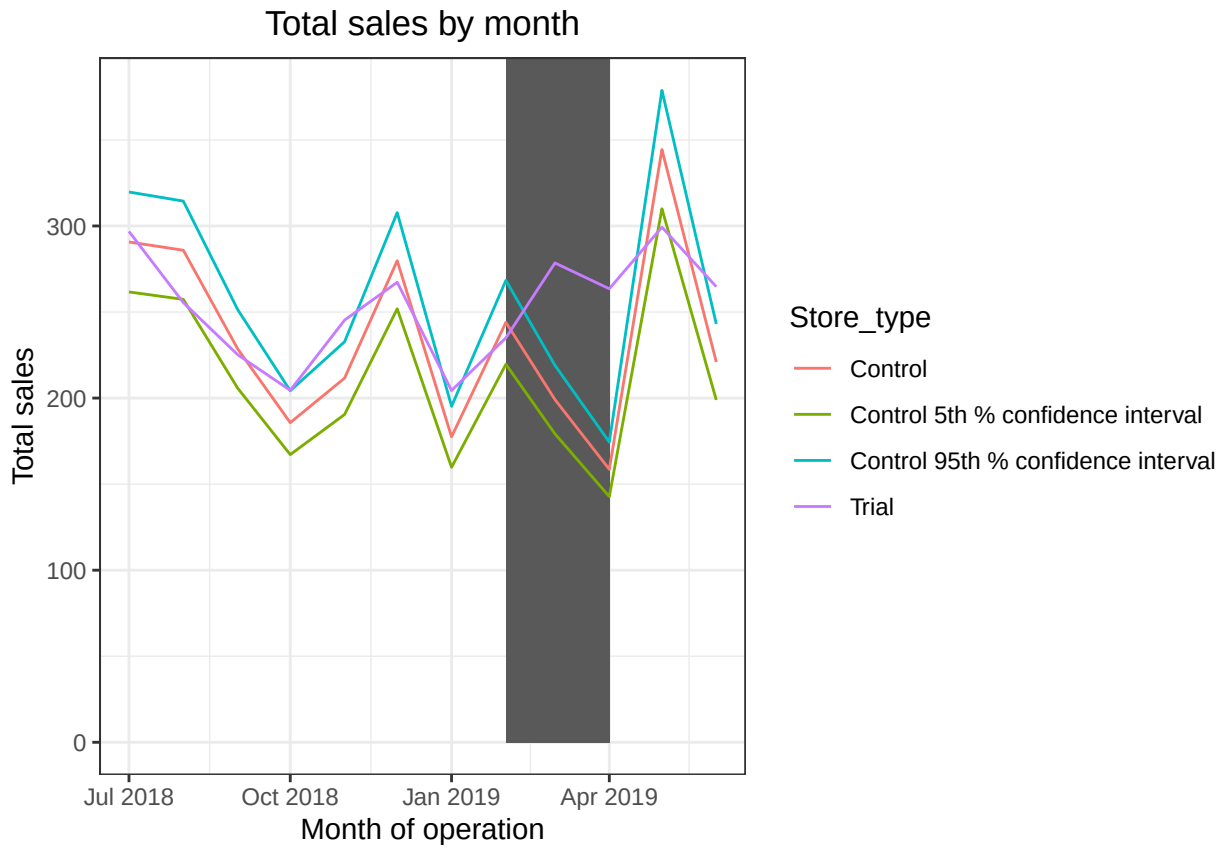
```r
# Calculate standard deviation of percentage differences for pre-trial period
stdDev <- sd(percentageDiff[YEARMONTH < 201902, percentageDiff])

# Note that there are 8 months in the pre-trial period, hence 8 - 1 = 7 degrees
 ↪  of freedom
degreesOfFreedom <- 7

# Test with a null hypothesis of there being 0 difference between trial and
 ↪  control stores
percentageDiff <- percentageDiff[
  , tValue := (percentageDiff - 0) / stdDev
][
  , TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH %% 100, 1,
 ↪  sep = "-"), "%Y-%m-%d")
][
  YEARMONTH < 201905 & YEARMONTH > 201901, .(TransactionMonth, tValue)
]
```

```r
#### Find the 95th percentile of the t distribution with the appropriate
qt(0.95, df = degreesOfFreedom)
```

```
## [1] 1.894579
```

```r
# Prepare measureOverTime data for sales
measureOverTimeSales <- measureOverTime

# Trial and control store total sales
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")
][
  Store_type %in% c("Trial", "Control")
```

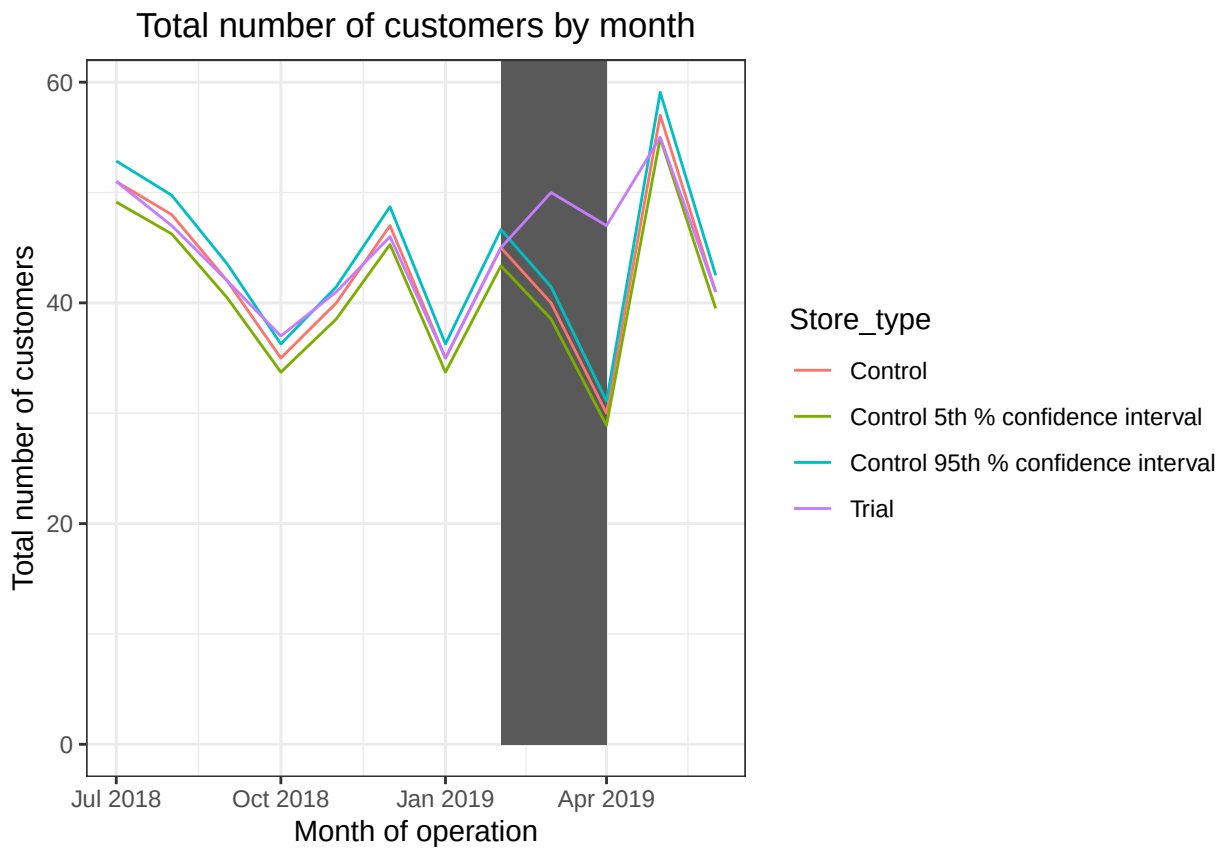
```
]

# 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"
]

# Combine all sales data for trial assessment
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")
```




```
# Calculate scaling factor for control customers
scalingFactorForControlCust <- preTrialMeasures[
  STORE_NBR == trial_store & YEARMONTH < 201902, sum(nCustomers)
] / preTrialMeasures[
  STORE_NBR == control_store & YEARMONTH < 201902, sum(nCustomers)
]

# Apply the scaling factor
measureOverTimeCusts <- measureOverTime

scaledControlCustomers <- measureOverTimeCusts[
  STORE_NBR == control_store,
][
  , controlCustomers := nCustomers * scalingFactorForControlCust
][
  , Store_type := ifelse(STORE_NBR == trial_store, "Trial",
                         ifelse(STORE_NBR == control_store, "Control", "Other
                          ↪  stores"))
]

# Calculate the percentage difference between scaled control customers and
 ↪  trial customers
percentageDiff <- merge(
  scaledControlCustomers[, .(YEARMONTH, controlCustomers)],
  measureOverTimeCusts[STORE_NBR == trial_store, .(nCustomers, YEARMONTH)],
  by = "YEARMONTH"
```

```
)[
  , percentageDiff := abs(controlCustomers - nCustomers) / controlCustomers
]
```

```
# Calculate standard deviation of percentage differences for pre-trial period
stdDev <- sd(percentageDiff[YEARMONTH < 201902, percentageDiff])

# Set degrees of freedom
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"
]

# Combine all customers data for trial assessment
trialAssessment <- rbind(pastCustomers, pastCustomers_Controls95,
 ↪  pastCustomers_Controls5)

# Plotting these in one nice graph
ggplot(trialAssessment, aes(TransactionMonth, nCusts, 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 number of customers", title = "Total
   ↪  number of customers by month")
```



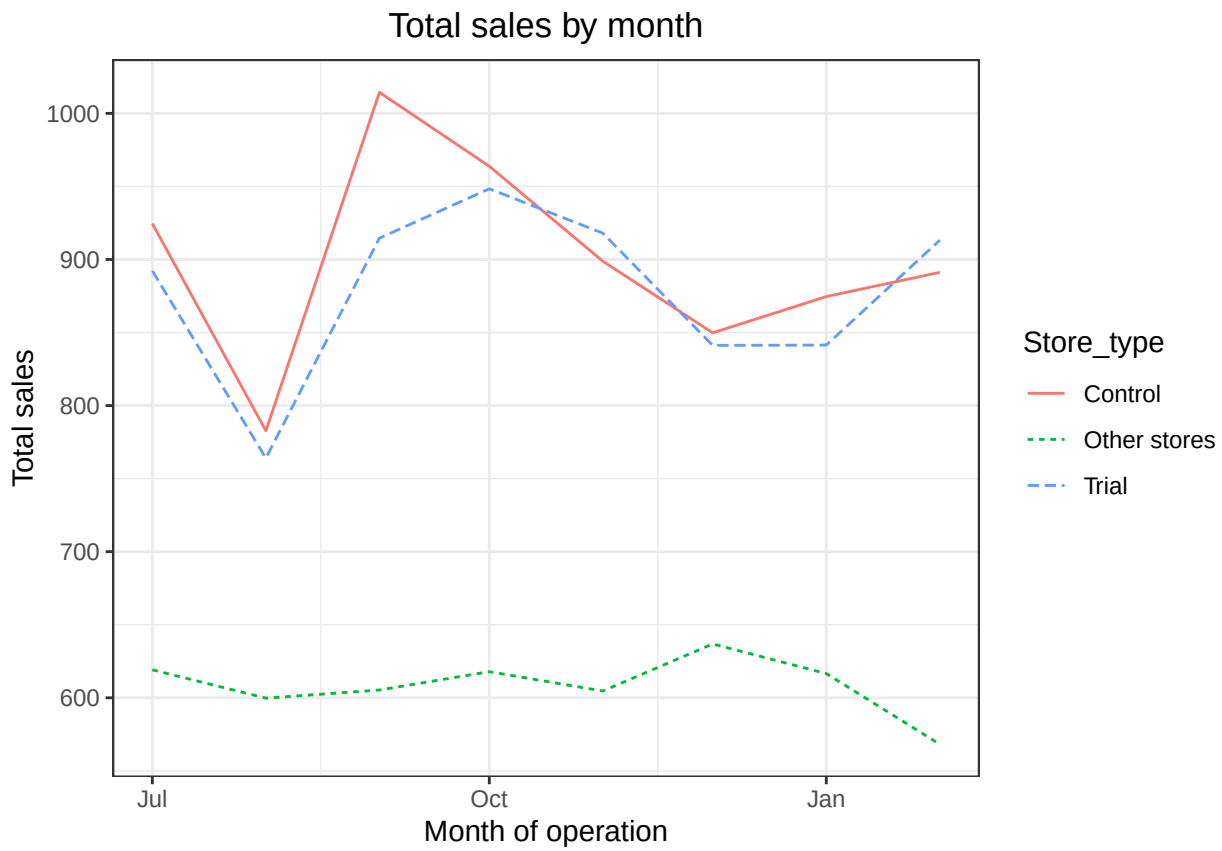
Trial store 86

```
# Create measureOverTime data table summarizing sales metrics
measureOverTime <- data[, .(
  totSales = sum(TOT_SALES),
  nCustomers = uniqueN(LYLTY_CARD_NBR),
  nTxnPerCust = uniqueN(TXN_ID) / uniqueN(LYLTY_CARD_NBR),
  nChipsPerTxn = sum(PROD_QTY) / uniqueN(TXN_ID),
  avgPricePerUnit = sum(TOT_SALES) / sum(PROD_QTY)
), by = c("STORE_NBR", "YEARMONTH")][order(STORE_NBR, YEARMONTH)]

# Set trial store number
trial_store <- 86

# Calculate correlation metrics
corr_nSales <- calculateCorrelation(preTrialMeasures, quote(totSales),
 ↪  trial_store)
corr_nCustomers <- calculateCorrelation(preTrialMeasures, quote(nCustomers),
 ↪  trial_store)

# Calculate magnitude distances
magnitude_nSales <- calculateMagnitudeDistance(preTrialMeasures,
 ↪  quote(totSales), trial_store)
magnitude_nCustomers <- calculateMagnitudeDistance(preTrialMeasures,
 ↪  quote(nCustomers), trial_store)
```

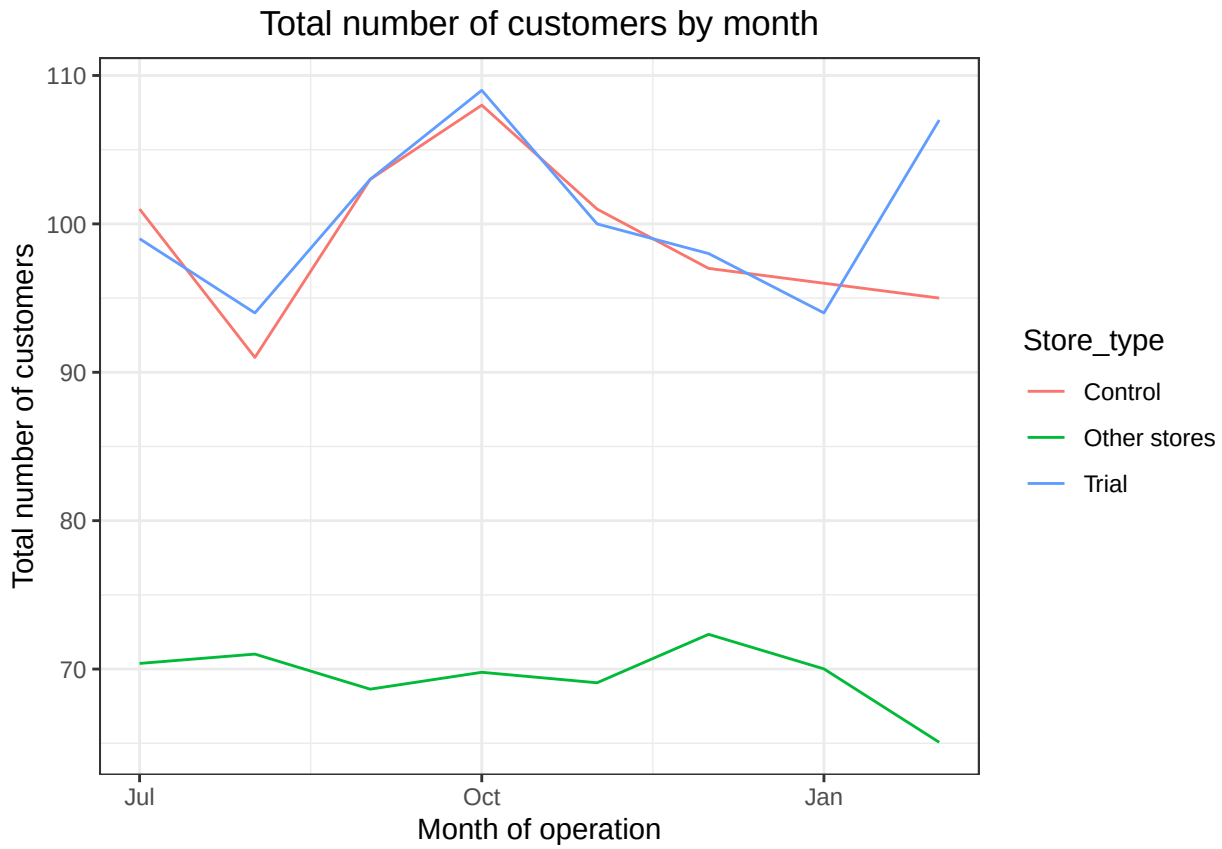
```
# Create a combined score composed of correlation and magnitude
corr_weight <- 0.5
score_nSales <- merge(corr_nSales, magnitude_nSales, by = c("Store1",
 ↪  "Store2"))[
  , scoreNSales := corr_measure * corr_weight + mag_measure * (1 - corr_weight)
]

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

# Combine scores across the drivers
score_Control <- merge(score_nSales, score_nCustomers, by = c("Store1",
 ↪  "Store2"))
score_Control[, finalControlScore := scoreNSales * 0.5 + scoreNCust * 0.5]

# Select control store based on the highest matching store (second highest
 ↪  ranked)
control_store <- score_Control[Store1 == trial_store,
][order(-finalControlScore)][2, Store2]

# Display the selected control store
control_store
```

```
## [1] 155
```

```
# Visual checks on trends based on the drivers
measureOverTimeSales <- measureOverTime

measureOverTimeSales[, YEARMONTH := as.numeric(YEARMONTH)]


# Create a data table summarizing total sales by store type and month
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, ]

# Plot total sales by month for different store types
ggplot(pastSales, aes(TransactionMonth, totSales, color = Store_type)) +
  geom_line(aes(linetype = Store_type)) +
  labs(x = "Month of operation", y = "Total sales", title = "Total sales by
   ↪  month")
```

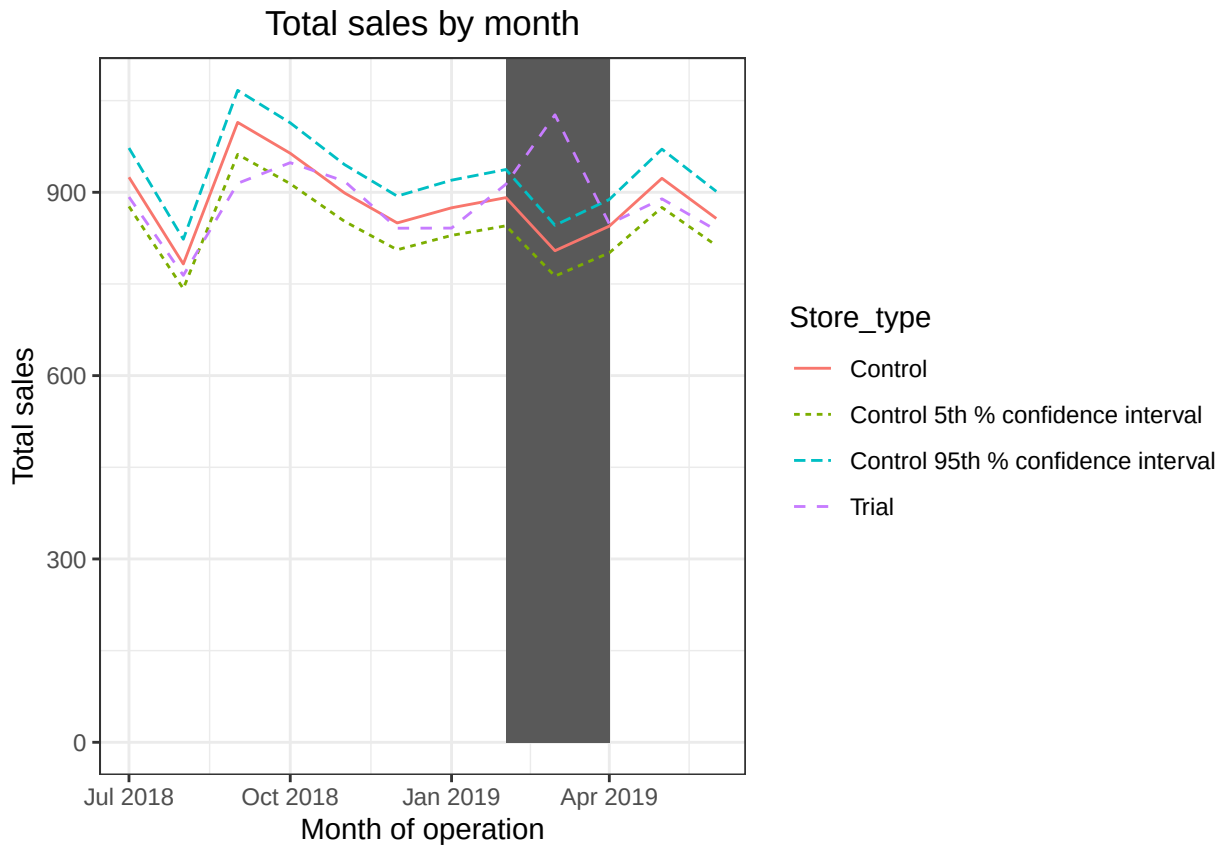



```
# Calculate the total number of customers over time
measureOverTimeCusts <- measureOverTime

# Create a data table summarizing the number of customers by store type and
 ↪  month
pastCustomers <- measureOverTimeCusts[, Store_type := ifelse(STORE_NBR ==
 ↪  trial_store, "Trial",
    ifelse(STORE_NBR == control_store, "Control", "Other stores"))
][, numberCustomers := mean(nCustomers), by = c("YEARMONTH", "Store_type")
][, TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH %% 100, 1,
 ↪  sep = "-"), "%Y-%m-%d")
][YEARMONTH < 201903, ]

# Plot the number of customers by month for different store types
ggplot(pastCustomers, aes(TransactionMonth, numberCustomers, color =
 ↪  Store_type)) +
  geom_line() +
  labs(x = "Month of operation", y = "Total number of customers", title = "Total
   ↪  number of customers by month")
```



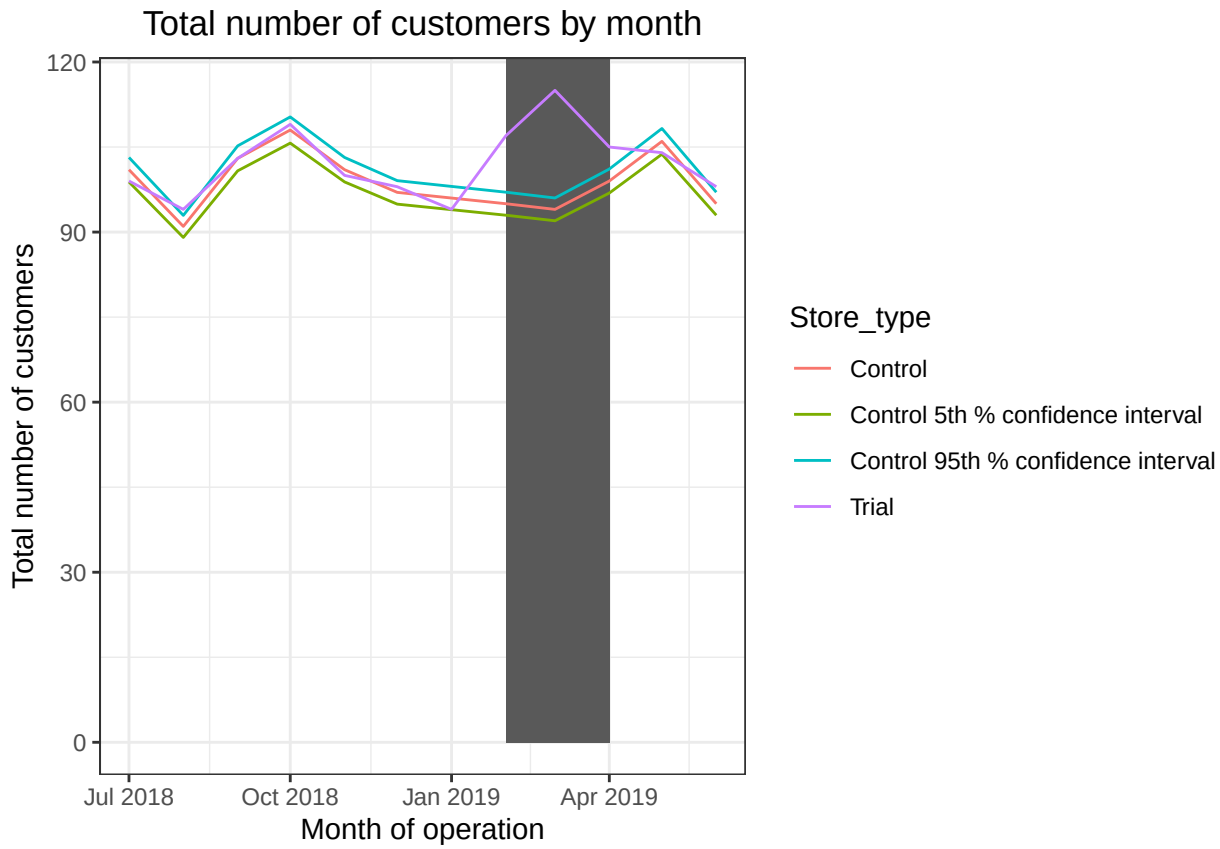

```
# Calculate the scaling factor for control sales based on pre-trial measures
scalingFactorForControlSales <- preTrialMeasures[STORE_NBR == trial_store &
  YEARMONTH < 201902, sum(totSales)] /
  preTrialMeasures[STORE_NBR == control_store & YEARMONTH < 201902,
 ↪  sum(totSales)]

# Apply the scaling factor to calculate scaled control sales
measureOverTimeSales <- measureOverTime
scaledControlSales <- measureOverTimeSales[STORE_NBR == control_store, ][,
  controlSales := totSales * scalingFactorForControlSales]

# Calculate the percentage difference between scaled control sales and trial
 ↪  sales
percentageDiff <- merge(
  scaledControlSales[, c("YEARMONTH", "controlSales")],
  measureOverTime[STORE_NBR == trial_store, c("totSales", "YEARMONTH")],
  by = "YEARMONTH"
)[, percentageDiff := abs(controlSales - totSales) / controlSales]

# Calculate the standard deviation and degrees of freedom
stdDev <- sd(percentageDiff[YEARMONTH < 201902, percentageDiff])
degreesOfFreedom <- 7

# Trial and control store total sales calculation
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")
][Store_type %in% c("Trial", "Control"), ]

# Control store 95th percentile calculation
pastSales_Controls95 <- pastSales[Store_type == "Control", ][,
  totSales := totSales * (1 + stdDev * 2)][, Store_type := "Control 95th %
 ↪  confidence interval"]

# Control store 5th percentile calculation
pastSales_Controls5 <- pastSales[Store_type == "Control", ][,
  totSales := totSales * (1 - stdDev * 2)][, Store_type := "Control 5th %
 ↪  confidence interval"]

# Combine past sales data for plotting
trialAssessment <- rbind(pastSales, pastSales_Controls95, pastSales_Controls5)

# Plotting total sales by month
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(aes(linetype = Store_type)) +
  labs(x = "Month of operation", y = "Total sales", title = "Total sales by
   ↪  month")
```




```
# Scale pre-trial control customers to match pre-trial trial store customers
scalingFactorForControlCust <- preTrialMeasures[STORE_NBR == trial_store &
  YEARMONTH < 201902, sum(nCustomers)] /
  preTrialMeasures[STORE_NBR == control_store & YEARMONTH < 201902,
 ↪  sum(nCustomers)]

# Apply the scaling factor
measureOverTimeCusts <- measureOverTime
scaledControlCustomers <- measureOverTimeCusts[STORE_NBR == control_store, ][,
  controlCustomers := nCustomers * scalingFactorForControlCust
][, Store_type := ifelse(STORE_NBR == trial_store, "Trial",
                         ifelse(STORE_NBR == control_store, "Control", "Other
                          ↪  stores"))]

# Calculate the percentage difference between scaled control customers and
 ↪  trial customers
percentageDiff <- merge(
  scaledControlCustomers[, c("YEARMONTH", "controlCustomers")],
  measureOverTimeCusts[STORE_NBR == trial_store, c("nCustomers", "YEARMONTH")],
  by = "YEARMONTH"
)[, percentageDiff := abs(controlCustomers - nCustomers) / controlCustomers]

# Calculate standard deviation for the percentage difference in the pre-trial
 ↪  period
stdDev <- sd(percentageDiff[YEARMONTH < 201902, percentageDiff])
```

```r
degreesOfFreedom <- 7

# Trial and control store number of customers calculation
pastCustomers <- measureOverTimeCusts[, nCusts := mean(nCustomers), by =
 ↪  c("YEARMONTH", "Store_type")
][Store_type %in% c("Trial", "Control"), ]

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

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

# Combine past customers data for plotting
trialAssessment <- rbind(pastCustomers, pastCustomers_Controls95,
 ↪  pastCustomers_Controls5)

# Plotting total number of customers by month
ggplot(trialAssessment, aes(TransactionMonth, nCusts, 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 number of customers", title = "Total
   ↪  number of customers by month")
```



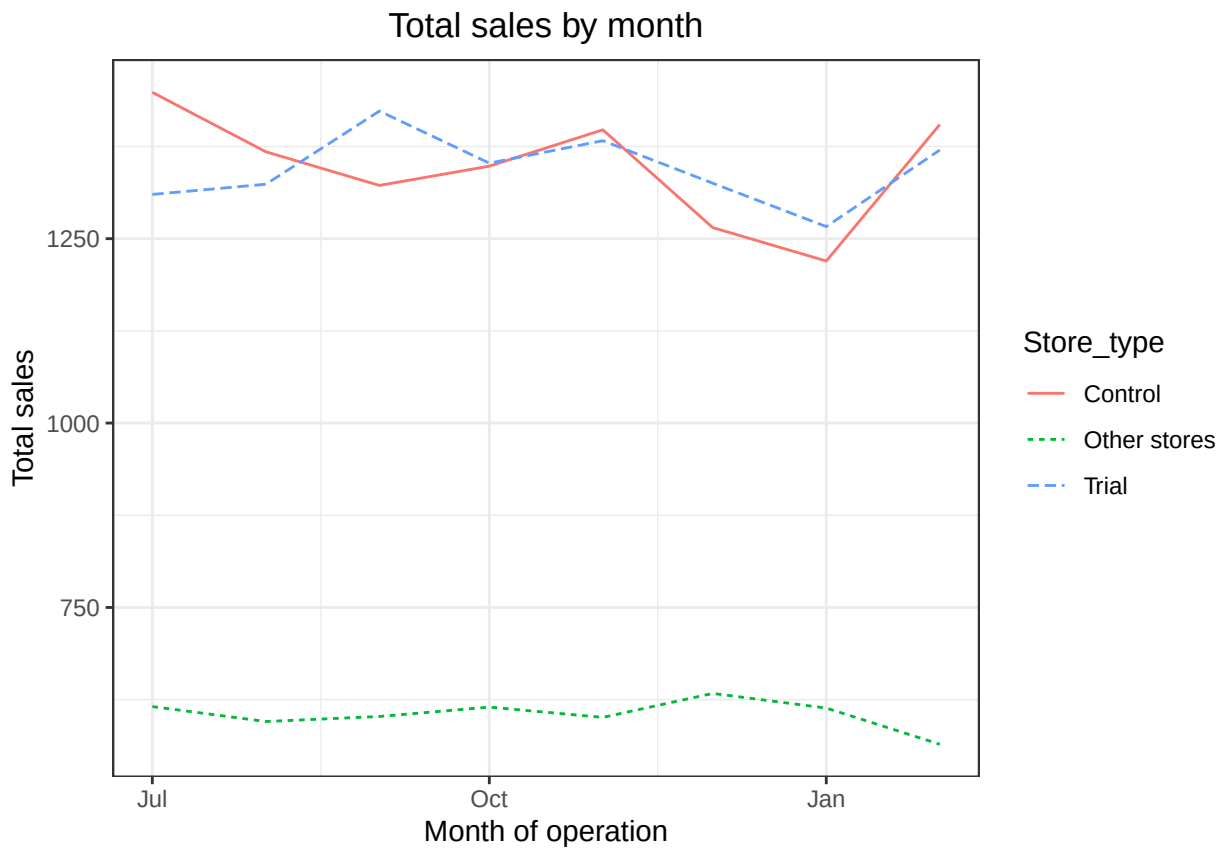
Trial store 88

```
# Calculate sales and customer metrics over time
measureOverTime <- data[, .(
  totSales = sum(TOT_SALES),
  nCustomers = uniqueN(LYLTY_CARD_NBR),
  nTxnPerCust = uniqueN(TXN_ID) / uniqueN(LYLTY_CARD_NBR),
  nChipsPerTxn = sum(PROD_QTY) / uniqueN(TXN_ID),
  avgPricePerUnit = sum(TOT_SALES) / sum(PROD_QTY)
), by = c("STORE_NBR", "YEARMONTH")][order(STORE_NBR, YEARMONTH)]

# Use the functions for calculating correlation
trial_store <- 88
corr_nSales <- calculateCorrelation(preTrialMeasures, quote(totSales),
 ↪  trial_store)
corr_nCustomers <- calculateCorrelation(preTrialMeasures, quote(nCustomers),
 ↪  trial_store)

# Use the functions for calculating magnitude
magnitude_nSales <- calculateMagnitudeDistance(preTrialMeasures,
 ↪  quote(totSales), trial_store)
magnitude_nCustomers <- calculateMagnitudeDistance(preTrialMeasures,
 ↪  quote(nCustomers), trial_store)
```

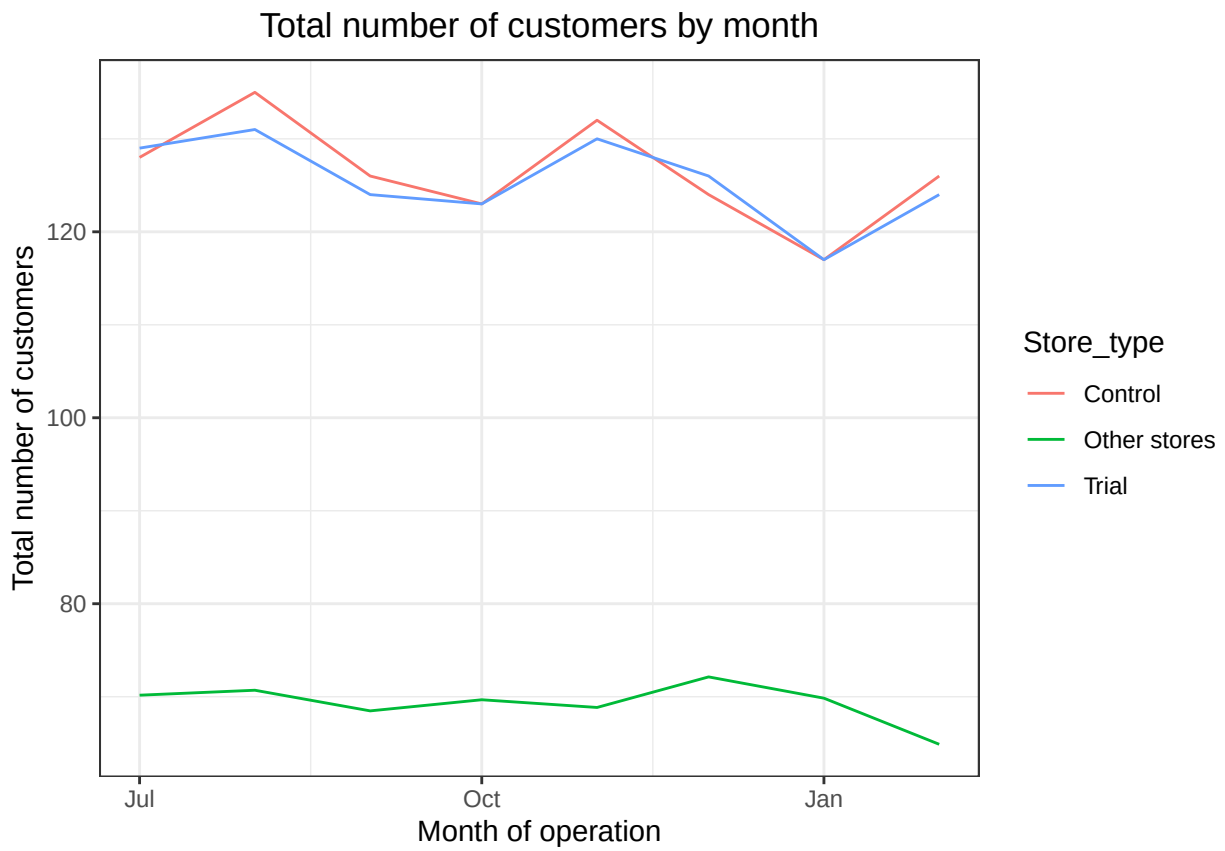
```r
# Create a combined score composed of correlation and magnitude
corr_weight <- 0.5
score_nSales <- merge(corr_nSales, magnitude_nSales, by = c("Store1",
 ↪  "Store2"))[,
  scoreNSales := corr_measure * corr_weight + mag_measure * (1 - corr_weight)]

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

# Combine scores across the drivers
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)
# Select control store for trial store 88
control_store <- score_Control[Store1 == trial_store,
 ↪  ][order(-finalControlScore)][2, Store2]

# Output the control store selected
control_store
```

```
## [1] 237
```

```r
# Prepare sales data for visualization
measureOverTimeSales <- measureOverTime

measureOverTimeSales[, YEARMONTH := as.numeric(YEARMONTH)]


# Summarize total sales by store type and month
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, ]

# Plot total sales by month for each store type
ggplot(pastSales, aes(TransactionMonth, totSales, color = Store_type)) +
  geom_line(aes(linetype = Store_type)) +
  labs(x = "Month of operation",
       y = "Total sales",
       title = "Total sales by month")
```



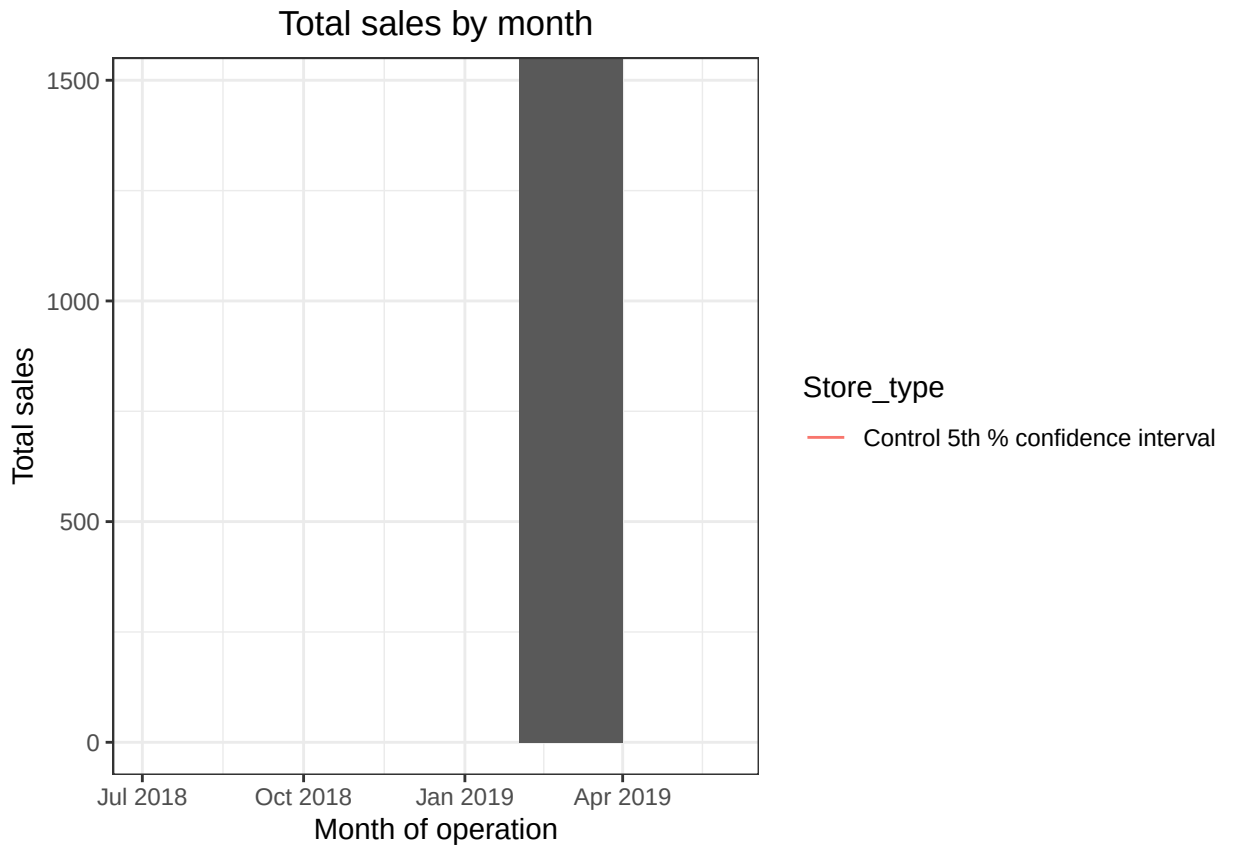

```
# Prepare customer data for visualization
measureOverTimeCusts <- measureOverTime

# Summarize number of customers by store type and month
pastCustomers <- measureOverTimeCusts[,
  Store_type := ifelse(STORE_NBR == trial_store, "Trial",
                       ifelse(STORE_NBR == control_store, "Control", "Other
                        ↪  stores"))][
  , numberCustomers := mean(nCustomers), by = c("YEARMONTH", "Store_type")][
  , TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH %% 100, 1,
 ↪  sep = "-"), "%Y-%m-%d")][
  YEARMONTH < 201903, ]

# Plot total number of customers by month for each store type
ggplot(pastCustomers, aes(TransactionMonth, numberCustomers, color =
 ↪  Store_type)) +
  geom_line() +
  labs(x = "Month of operation",
       y = "Total number of customers",
       title = "Total number of customers by month")
```




```
# 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]

# Calculate the percentage difference between scaled control sales and trial
 ↪  sales
percentageDiff <- merge(
  scaledControlSales[, .(YEARMONTH, controlSales)],
  measureOverTime[STORE_NBR == trial_store, .(totSales, YEARMONTH)],
  by = "YEARMONTH"
)[, percentageDiff := abs(controlSales - totSales) / controlSales]

# Standard deviation of the percentage difference in the pre-trial period
stdDev <- sd(percentageDiff[YEARMONTH < 201902, percentageDiff])
degreesOfFreedom <- 7

# Trial and control store total sales
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")][
  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"]

# Combine trial and control sales data for plotting
trialAssessment <- rbind(pastSales, pastSales_Controls95, pastSales_Controls5)

# Plotting total sales by month
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(aes(linetype = Store_type)) +
  labs(x = "Month of operation",
       y = "Total sales",
       title = "Total sales by month")
```

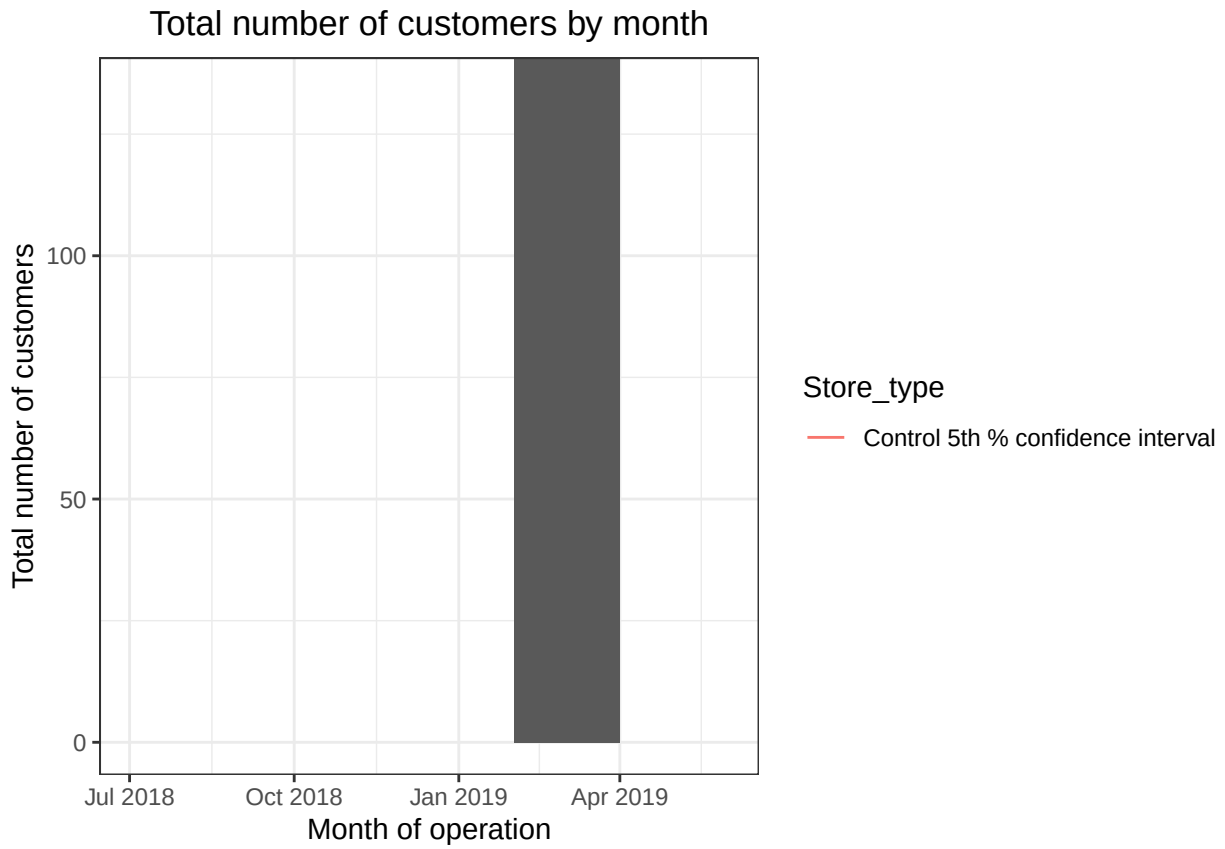
```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```




```
# Scale pre-trial control customers to match pre-trial trial store customers
scalingFactorForControlCust <- preTrialMeasures[STORE_NBR == trial_store &
                                                  YEARMONTH < 201902,
 ↪  sum(nCustomers)] /
                                 preTrialMeasures[STORE_NBR == control_store &
                                                    YEARMONTH < 201902, sum(nCustomers)]

# Apply the scaling factor
measureOverTimeCusts <- measureOverTime
scaledControlCustomers <- measureOverTimeCusts[STORE_NBR == control_store,
                                                 controlCustomers := nCustomers *
 ↪  scalingFactorForControlCust][
                                               , Store_type := ifelse(STORE_NBR ==
 ↪  trial_store,
                                                                      "Trial",
                                                           ifelse(STORE_NBR == con-
                                                                    ↪  trol_store,
                                                                    ↪
                                                                  "Control",
                                                         "Other stores"))]

# Calculate the percentage difference between scaled control customers and
 ↪  trial customers
percentageDiff <- merge(
  scaledControlCustomers[, .(YEARMONTH, controlCustomers)],
  measureOverTime[STORE_NBR == trial_store, .(nCustomers, YEARMONTH)],
```

```
  by = "YEARMONTH"
)[, percentageDiff := abs(controlCustomers - nCustomers) / controlCustomers]

# Standard deviation of the percentage difference in the pre-trial period
stdDev <- sd(percentageDiff[YEARMONTH < 201902, percentageDiff])
degreesOfFreedom <- 7  # 8 months in the pre-trial period; hence 8 - 1 = 7
 ↪  degrees of freedom

# 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"]

# Combine trial and control customer data for plotting
trialAssessment <- rbind(pastCustomers, pastCustomers_Controls95,
 ↪  pastCustomers_Controls5)

# Plotting total number of customers by month
ggplot(trialAssessment, aes(TransactionMonth, nCusts, 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 number of customers",
       title = "Total number of customers by month")
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```



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.