# Quantium VI Task 2

Rolly Mougoue

2022-04-28

## Task 2 : Retail Strategy and Analytics

### Load required libraries and datasets

We'll start with setting up our working environment by installing and loading all the necessary packages (Tidyverse, Lubridate, dplyr, pad...) and setting the working directory for easy upload of our datasets

### Set themes for plots

```
theme_set(theme_bw())
theme_update(plot.title = element_text(hjust = 0.5))
Data <- data.table(Data)
```

```
show(Data)
```

```
##         LYLTY_CARD_NBR       DATE STORE_NBR TXN_ID PROD_NBR
##      1:           1000 2018-10-17         1      1        5
##      2:           1002 2018-09-16         1      2       58
##      3:           1003 2019-03-07         1      3       52
##      4:           1003 2019-03-08         1      4      106
##      5:           1004 2018-11-02         1      5       96
##     ---
## 264830:        2370701 2018-12-08        88 240378       24
## 264831:        2370751 2018-10-01        88 240394       60
## 264832:        2370961 2018-10-24        88 240480       70
## 264833:        2370961 2018-10-27        88 240481       65
## 264834:        2373711 2018-12-14        88 241815       16
##                                         PROD_NAME PROD_QTY TOT_SALES PACK_SIZE
##      1:   Natural Chip        Compny SeaSalt175g        2       6.0       175
##      2:    Red Rock Deli Chikn&Garlic Aioli 150g        1       2.7       150
##      3:    Grain Waves Sour    Cream&Chives 210G        1       3.6       210
##      4:   Natural ChipCo      Hony Soy Chckn175g        1       3.0       175
##      5:           WW Original Stacked Chips 160g        1       1.9       160
##     ---
## 264830:    Grain Waves         Sweet Chilli 210g        2       7.2       210
## 264831:     Kettle Tortilla ChpsFeta&Garlic 150g        2       9.2       150
## 264832:  Tyrrells Crisps     Lightly Salted 165g        2       8.4       165
## 264833: Old El Paso Salsa   Dip Chnky Tom Ht300g        2      10.2       300
```

```
## 264834: Smiths Crinkle Chips Salt & Vinegar 330g        2      11.4       330
##              BRAND            LIFESTAGE PREMIUM_CUSTOMER
##      1:    NATURAL YOUNG SINGLES/COUPLES          Premium
##      2:        RRD YOUNG SINGLES/COUPLES       Mainstream
##      3:    GRNWVES        YOUNG FAMILIES           Budget
##      4:    NATURAL        YOUNG FAMILIES           Budget
##      5: WOOLWORTHS OLDER SINGLES/COUPLES       Mainstream
##     ---
## 264830:    GRNWVES        YOUNG FAMILIES       Mainstream
## 264831:     KETTLE        YOUNG FAMILIES          Premium
## 264832:   TYRRELLS        OLDER FAMILIES           Budget
## 264833:        OLD        OLDER FAMILIES           Budget
## 264834:     SMITHS YOUNG SINGLES/COUPLES       Mainstream
```

### 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.

```
#### Calculate these measures over time for each store
## Add a new month ID column

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

#### Next, we define the measure calculations

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)]
```

```
measureOverTime
```

```
##       STORE_NBR YEARMONTH totSales nCustomers nTxnPerCust nChipsPerTxn
##    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
##   ---
## 3165:       272    201902    395.5         45    1.066667     1.895833
## 3166:       272    201903    442.3         50    1.060000     1.905660
## 3167:       272    201904    445.1         54    1.018519     1.909091
## 3168:       272    201905    314.6         34    1.176471     1.775000
## 3169:       272    201906    312.1         34    1.088235     1.891892
##       avgPricePerUnit
##    1:        3.337097
##    2:        3.261111
```

```
##    3:        3.717333
##    4:        3.243103
##    5:        3.378947
##   ---
## 3165:        4.346154
## 3166:        4.379208
## 3167:        4.239048
## 3168:        4.430986
## 3169:        4.458571
```

```
#### Filter to the pre-trial period and stores with full observation periods
storesWithFullObs <- unique(measureOverTime[, .N, STORE_NBR][N == 12, STORE_NBR])
storesWithFullObs
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  12  13  14  15  16  17  18  19
##  [19]  20  21  22  23  24  25  26  27  28  29  30  32  33  34  35  36  37  38
##  [37]  39  40  41  42  43  45  46  47  48  49  50  51  52  53  54  55  56  57
##  [55]  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75
##  [73]  77  78  79  80  81  82  83  84  86  87  88  89  90  91  93  94  95  96
##  [91]  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114
## [109] 115 116 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133
## [127] 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151
## [145] 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169
## [163] 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187
## [181] 188 189 190 191 192 194 195 196 197 198 199 200 201 202 203 204 205 207
## [199] 208 209 210 212 213 214 215 216 217 219 220 221 222 223 224 225 226 227
## [217] 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245
## [235] 246 247 248 249 250 251 253 254 255 256 257 258 259 260 261 262 263 264
## [253] 265 266 267 268 269 270 271 272
```

```
preTrialMeasures <- measureOverTime[YEARMONTH < 201902 & STORE_NBR %in% storesWithFullObs, ]
preTrialMeasures
```

```
##       STORE_NBR YEARMONTH totSales nCustomers nTxnPerCust nChipsPerTxn
##    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
##   ---
## 1816:       272    201809    304.7         32    1.125000     1.972222
## 1817:       272    201810    430.6         44    1.136364     1.980000
## 1818:       272    201811    376.2         41    1.097561     1.933333
## 1819:       272    201812    403.9         47    1.000000     1.893617
## 1820:       272    201901    423.0         46    1.086957     1.920000
##       avgPricePerUnit
##    1:        3.337097
##    2:        3.261111
##    3:        3.717333
##    4:        3.243103
##    5:        3.378947
##   ---
## 1816:        4.291549
```

```
## 1817:        4.349495
## 1818:        4.324138
## 1819:        4.538202
## 1820:        4.406250
```

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.

```
#### Let's define inputTable as a metric table with potential comparison stores, metricCol as the store

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_NI
                                                                                                 eval(metr

    calcCorrTable <- rbind(calcCorrTable, calculatedMeasure)
  }

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.

```
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(metric
                                                   - inputTable[STORE_NBR == i,eval(metricCol)])
                                   )
    calcDistTable <- rbind(calcDistTable, calculatedMeasure)
  }

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

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

```r
  finalDistTable <- distTable[, .(mag_measure = mean(magnitudeMeasure)), 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.

```r
#### Calculate correlations against store 77 using total sales and number of customers.
trial_store <- 77

corr_nSales <- calculateCorrelation(preTrialMeasures, quote(totSales), trial_store)
corr_nSales[order(-corr_measure)]
```

```
##      Store1 Store2 corr_measure
##   1:     77     77    1.0000000
##   2:     77     71    0.9141060
##   3:     77    233    0.9037742
##   4:     77    119    0.8676644
##   5:     77     17    0.8426684
##  ---
## 256:     77    158   -0.7093194
## 257:     77     24   -0.7181123
## 258:     77    244   -0.7745129
## 259:     77     75   -0.8067514
## 260:     77    186   -0.8202139
```

```r
corr_nCustomers <- calculateCorrelation(preTrialMeasures, quote(nCustomers), trial_store)
corr_nCustomers[order(-corr_measure)]
```

```
##      Store1 Store2 corr_measure
##   1:     77     77    1.0000000
##   2:     77    233    0.9903578
##   3:     77    119    0.9832666
##   4:     77    254    0.9162084
##   5:     77    113    0.9013480
##  ---
## 256:     77    102   -0.6525273
## 257:     77    147   -0.6569333
## 258:     77    169   -0.6663911
## 259:     77     54   -0.7606047
## 260:     77      9   -0.7856990
```

```r
#### Then, use the functions for calculating magnitude.
magnitude_nSales <- calculateMagnitudeDistance(preTrialMeasures, quote(totSales), trial_store)
magnitude_nSales[order(-mag_measure)]
```

```
##      Store1 Store2 mag_measure
##   1:     77     77  1.00000000
##   2:     77    233  0.98526489
##   3:     77    255  0.97672145
```

```
##   4:     77     53  0.97542233
##   5:     77    188  0.97517706
##  ---
## 256:     77     58  0.17395834
## 257:     77    165  0.16682996
## 258:     77    237  0.14886586
## 259:     77     88  0.14760746
## 260:     77    226  0.05985349
```

```
magnitude_nCustomers <- calculateMagnitudeDistance(preTrialMeasures, quote(nCustomers), trial_store)
magnitude_nCustomers[order(-mag_measure)]
```

```
##      Store1 Store2 mag_measure
##   1:     77     77  1.00000000
##   2:     77    233  0.99277331
##   3:     77     41  0.97463924
##   4:     77    111  0.96606414
##   5:     77    115  0.96591604
##  ---
## 256:     77     40  0.17065304
## 257:     77     58  0.15547195
## 258:     77     88  0.14457580
## 259:     77    237  0.13640397
## 260:     77    226  0.04279467
```

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.

```
corr_weight <- 0.5
score_nSales <- merge(corr_nSales, magnitude_nSales, by =
                        c("Store1", "Store2"))[, scoreNSales := (corr_measure + mag_measure)/2]
score_nSales[order(-scoreNSales)]
```

```
##      Store1 Store2 corr_measure mag_measure scoreNSales
##   1:     77     77    1.0000000   1.0000000  1.00000000
##   2:     77    233    0.9037742   0.9852649  0.94451954
##   3:     77     41    0.7832319   0.9651401  0.87418598
##   4:     77     50    0.7638658   0.9731293  0.86849757
##   5:     77     17    0.8426684   0.8806882  0.86167830
##  ---
## 256:     77    247   -0.6310496   0.5263807 -0.05233446
## 257:     77     24   -0.7181123   0.5908516 -0.06363035
## 258:     77    201   -0.4109081   0.2809523 -0.06497786
## 259:     77     55   -0.6667816   0.4693768 -0.09870241
## 260:     77     75   -0.8067514   0.3061880 -0.25028171
```

```
score_nCustomers <- merge(corr_nCustomers, magnitude_nCustomers, by =
                            c("Store1", "Store2"))[, scoreNCust := (corr_measure + mag_measure)/2]
score_nCustomers[order(-scoreNCust)]
```

```
##      Store1 Store2 corr_measure mag_measure  scoreNCust
##   1:     77     77    1.0000000   1.0000000  1.00000000
##   2:     77    233    0.9903578   0.9927733  0.99156555
##   3:     77    254    0.9162084   0.9371312  0.92666979
##   4:     77     41    0.8442195   0.9746392  0.90942936
##   5:     77     84    0.8585712   0.9241818  0.89137652
##  ---
## 256:     77    147   -0.6569333   0.4991028 -0.07891525
## 257:     77    247   -0.6210342   0.4278646 -0.09658482
## 258:     77    227   -0.6237974   0.3923204 -0.11573851
## 259:     77     75   -0.5907354   0.3360498 -0.12734284
## 260:     77    102   -0.6525273   0.3968462 -0.12784056
```

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.

```
score_Control <- merge(score_nSales, score_nCustomers, by =
                         c("Store1", "Store2"))

score_Control[, finalControlScore := scoreNSales * 0.5 + scoreNCust * 0.5]

score_Control[order(-finalControlScore)]
```

```
##      Store1 Store2 corr_measure.x mag_measure.x scoreNSales corr_measure.y
##   1:     77     77      1.0000000     1.0000000  1.00000000      1.0000000
##   2:     77    233      0.9037742     0.9852649  0.94451954      0.9903578
##   3:     77     41      0.7832319     0.9651401  0.87418598      0.8442195
##   4:     77     17      0.8426684     0.8806882  0.86167830      0.7473078
##   5:     77    254      0.5771085     0.9227714  0.74993992      0.9162084
##  ---
## 256:     77     55     -0.6667816     0.4693768 -0.09870241     -0.3954735
## 257:     77    138     -0.5851740     0.4913360 -0.04691903     -0.5348775
## 258:     77    247     -0.6310496     0.5263807 -0.05233446     -0.6210342
## 259:     77    102     -0.5508337     0.4885443 -0.03114471     -0.6525273
## 260:     77     75     -0.8067514     0.3061880 -0.25028171     -0.5907354
##      mag_measure.y   scoreNCust finalControlScore
##   1:     1.0000000  1.000000000        1.00000000
##   2:     0.9927733  0.991565547        0.96804254
##   3:     0.9746392  0.909429365        0.89180767
##   4:     0.9624953  0.854901530        0.85828992
##   5:     0.9371312  0.926669792        0.83830486
##  ---
## 256:     0.3797372 -0.007868115       -0.05328526
## 257:     0.3874739 -0.073701805       -0.06031042
## 258:     0.4278646 -0.096584823       -0.07445964
## 259:     0.3968462 -0.127840565       -0.07949264
## 260:     0.3360498 -0.127342842       -0.18881227
```

The store with the highest score is then selected as the control store since it is most similar to the trial store. From our results we can see that 233 can be selected as control store. Lets confirm that

```r
control_store <- score_Control[Store1 == trial_store, ][order(-finalControlScore)][2, Store2]
control_store
```

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

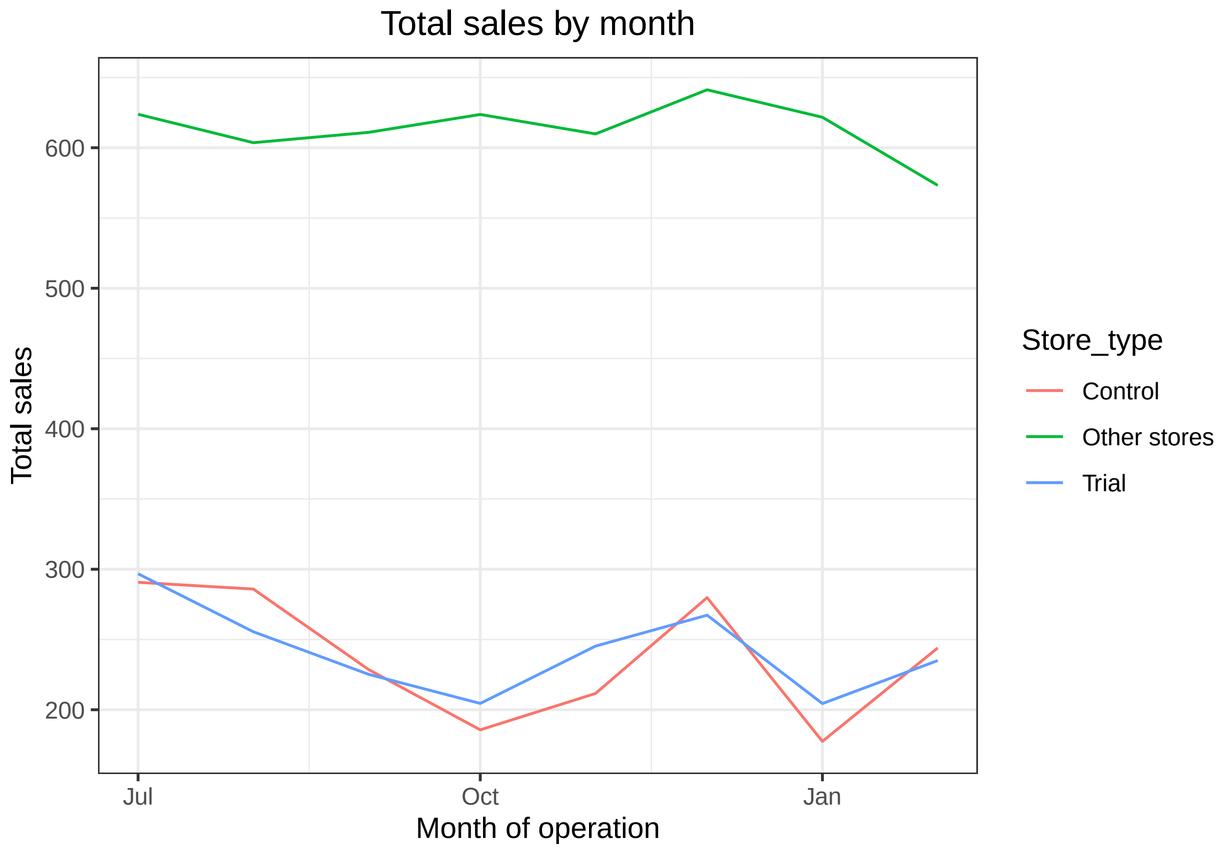
## Including Plots

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
pastSales
```

```
##       STORE_NBR YEARMONTH totSales nCustomers nTxnPerCust nChipsPerTxn
##    1:         1    201807 623.8174         49    1.061224     1.192308
##    2:         1    201808 603.6002         42    1.023810     1.255814
##    3:         1    201809 610.9473         59    1.050847     1.209677
##    4:         1    201810 623.6711         44    1.022727     1.288889
##    5:         1    201811 609.8351         46    1.021739     1.212766
##   ---
## 2108:       272    201810 623.6711         44    1.136364     1.980000
## 2109:       272    201811 609.8351         41    1.097561     1.933333
## 2110:       272    201812 641.2502         47    1.000000     1.893617
## 2111:       272    201901 621.6874         46    1.086957     1.920000
## 2112:       272    201902 573.2290         45    1.066667     1.895833
##       avgPricePerUnit   Store_type TransactionMonth
##    1:        3.337097 Other stores       2018-07-01
##    2:        3.261111 Other stores       2018-08-01
##    3:        3.717333 Other stores       2018-09-01
##    4:        3.243103 Other stores       2018-10-01
##    5:        3.378947 Other stores       2018-11-01
##   ---
## 2108:        4.349495 Other stores       2018-10-01
## 2109:        4.324138 Other stores       2018-11-01
## 2110:        4.538202 Other stores       2018-12-01
## 2111:        4.406250 Other stores       2019-01-01
## 2112:        4.346154 Other stores       2019-02-01
```

```r
ggplot(pastSales, aes(TransactionMonth, totSales, color = Store_type)) +
 geom_line() +labs(x = "Month of operation", y = "Total sales", title = "Total sales by month")
```



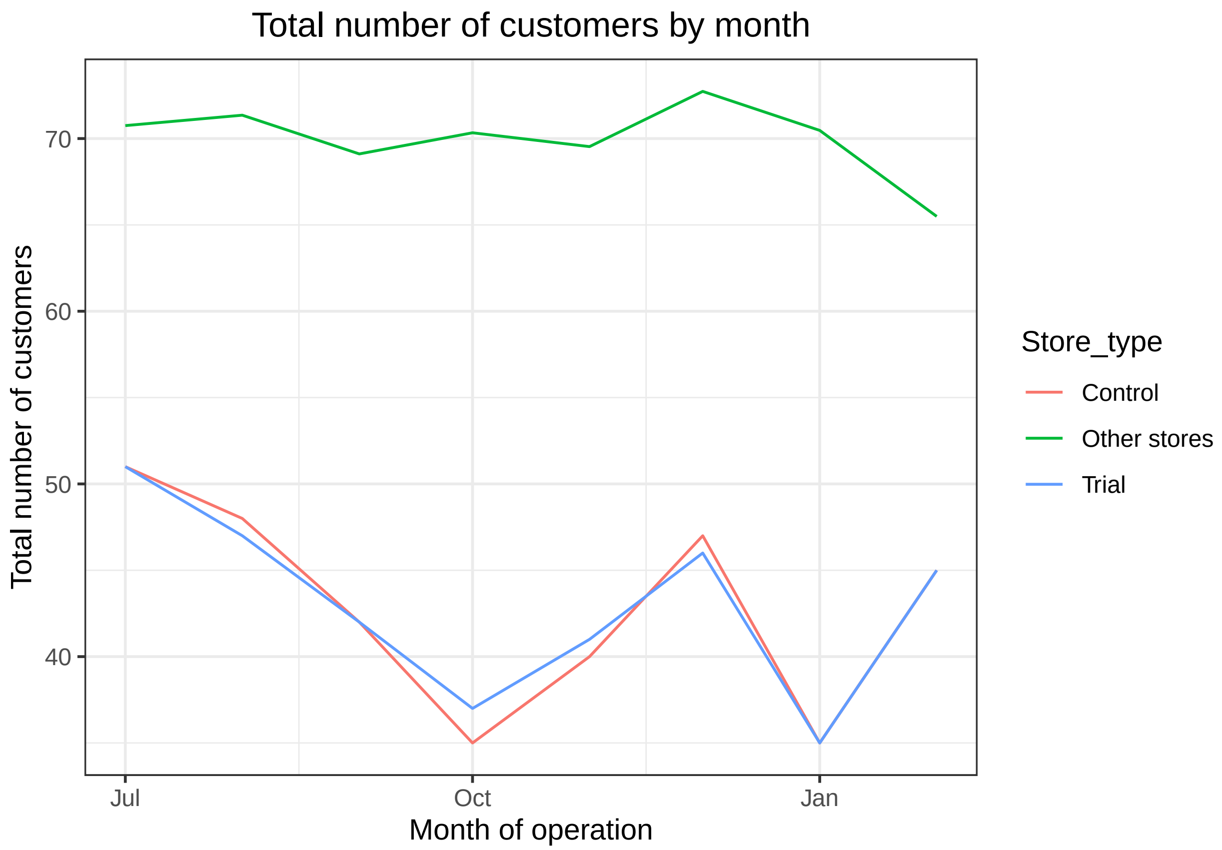

Next, number of customers.

```
unique(pastCusts)
```

```
##       STORE_NBR YEARMONTH totSales nCustomers nTxnPerCust nChipsPerTxn
##    1:         1    201807 623.8174         49    1.061224     1.192308
##    2:         1    201808 603.6002         42    1.023810     1.255814
##    3:         1    201809 610.9473         59    1.050847     1.209677
##    4:         1    201810 623.6711         44    1.022727     1.288889
##    5:         1    201811 609.8351         46    1.021739     1.212766
##   ---
## 2108:       272    201810 623.6711         44    1.136364     1.980000
## 2109:       272    201811 609.8351         41    1.097561     1.933333
## 2110:       272    201812 641.2502         47    1.000000     1.893617
## 2111:       272    201901 621.6874         46    1.086957     1.920000
## 2112:       272    201902 573.2290         45    1.066667     1.895833
##       avgPricePerUnit   Store_type TransactionMonth numberCustomers
##    1:        3.337097 Other stores       2018-07-01        70.75000
##    2:        3.261111 Other stores       2018-08-01        71.35249
##    3:        3.717333 Other stores       2018-09-01        69.11069
##    4:        3.243103 Other stores       2018-10-01        70.33460
##    5:        3.378947 Other stores       2018-11-01        69.53435
##   ---
## 2108:        4.349495 Other stores       2018-10-01        70.33460
## 2109:        4.324138 Other stores       2018-11-01        69.53435
## 2110:        4.538202 Other stores       2018-12-01        72.73180
```

```
## 2111:        4.406250 Other stores        2019-01-01        70.47126
## 2112:        4.346154 Other stores        2019-02-01        65.49237
```

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




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[ST
                                                   YEARMONTH < 201902, sum(totSales)]

#### Apply the scaling factor
measureOverTimeSales <- measureOverTime
scaledControlSales <- measureOverTimeSales[STORE_NBR == control_store, ][ , controlSales := totSales * s
```

```
scaledControlSales

measureOverTime[STORE_NBR == trial_store]
```

```
##     STORE_NBR YEARMONTH totSales nCustomers nTxnPerCust nChipsPerTxn
##   1:        77    201807    296.8         51    1.078431     1.527273
```

```
##  2:        77    201808    255.5         47    1.021277     1.541667
##  3:        77    201809    225.2         42    1.047619     1.590909
##  4:        77    201810    204.5         37    1.027027     1.368421
##  5:        77    201811    245.3         41    1.073171     1.522727
##  6:        77    201812    267.3         46    1.043478     1.500000
##  7:        77    201901    204.4         35    1.114286     1.666667
##  8:        77    201902    235.0         45    1.000000     1.644444
##  9:        77    201903    278.5         50    1.100000     1.490909
## 10:        77    201904    263.5         47    1.021277     1.625000
## 11:        77    201905    299.3         55    1.018182     1.500000
## 12:        77    201906    264.7         41    1.024390     1.666667
##     avgPricePerUnit Store_type TransactionMonth numberCustomers
##  1:        3.533333      Trial       2018-07-01              51
##  2:        3.452703      Trial       2018-08-01              47
##  3:        3.217143      Trial       2018-09-01              42
##  4:        3.932692      Trial       2018-10-01              37
##  5:        3.661194      Trial       2018-11-01              41
##  6:        3.712500      Trial       2018-12-01              46
##  7:        3.144615      Trial       2019-01-01              35
##  8:        3.175676      Trial       2019-02-01              45
##  9:        3.396341      Trial       2019-03-01              50
## 10:        3.378205      Trial       2019-04-01              47
## 11:        3.563095      Trial       2019-05-01              55
## 12:        3.781429      Trial       2019-06-01              41
```

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.

```
#### 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]
```

```
percentageDiff # between control store sales and trial store sales
```

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 differencein the 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
```

```
percentageDiff[, tValue := (percentageDiff - 0)/stdDev
               ][, TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH %% 100, 1,
                                                     sep = "-"), "%Y-%m-%d")
][YEARMONTH < 201905 & YEARMONTH > 201901, .(TransactionMonth,tValue)]
```

```
##    TransactionMonth    tValue
```

```
## 1:       2019-02-01  1.183534
## 2:       2019-03-01  7.339116
## 3:       2019-04-01 12.476373
```

```
#### Find the 95th percentile of the t distribution with the appropriate
#### degrees of freedom to compare against
qt(0.95, df = degreesOfFreedom)
```

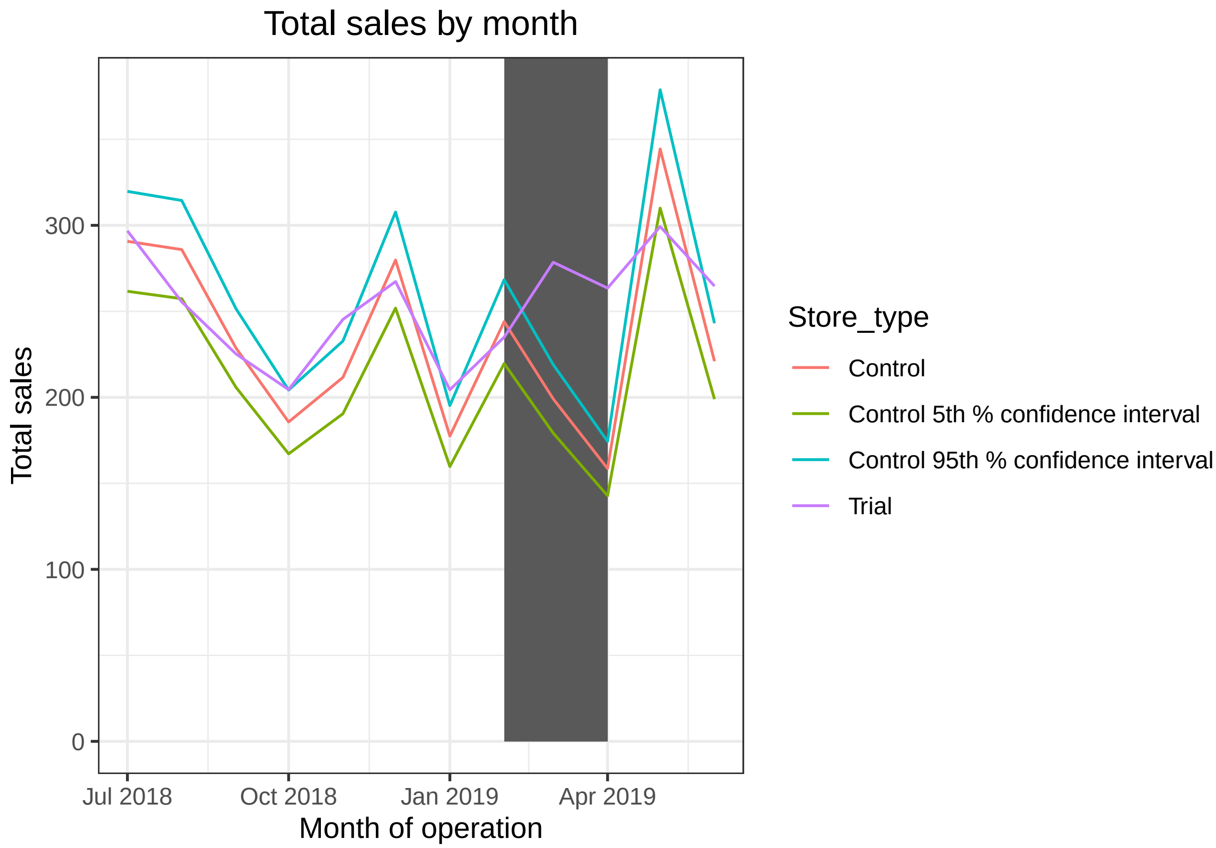
```
## [1] 1.894579
```

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
pastSales <- measureOverTimeSales[, Store_type := ifelse(STORE_NBR == trial_store, "Trial", ifelse(STORE
             control_store, "Control", "Other stores"))][, totSales := mean(totSales), by = c("YEARMONTH
             ][, TransactionMonth := as.Date(paste(YEARMONTH %/% 100, YEARMONTH %% 100, 1, sep = "-"),
             ][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)
```

```
ggplot(trialAssessment, aes(TransactionMonth, totSales, color = Store_type)) +
  geom_rect(data = trialAssessment[ YEARMONTH < 201905 & YEARMONTH > 201901 ,], aes(xmin = min(Transact
 geom_line() +
 labs(x = "Month of operation", y = "Total sales", title = "Total sales by month")
```




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.

```
#### Scale pre-trial control sales to match pre-trial trial store sales
scalingFactorForControlCust <- preTrialMeasures[STORE_NBR == trial_store & YEARMONTH < 201902, sum(nCus
  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"))]
```

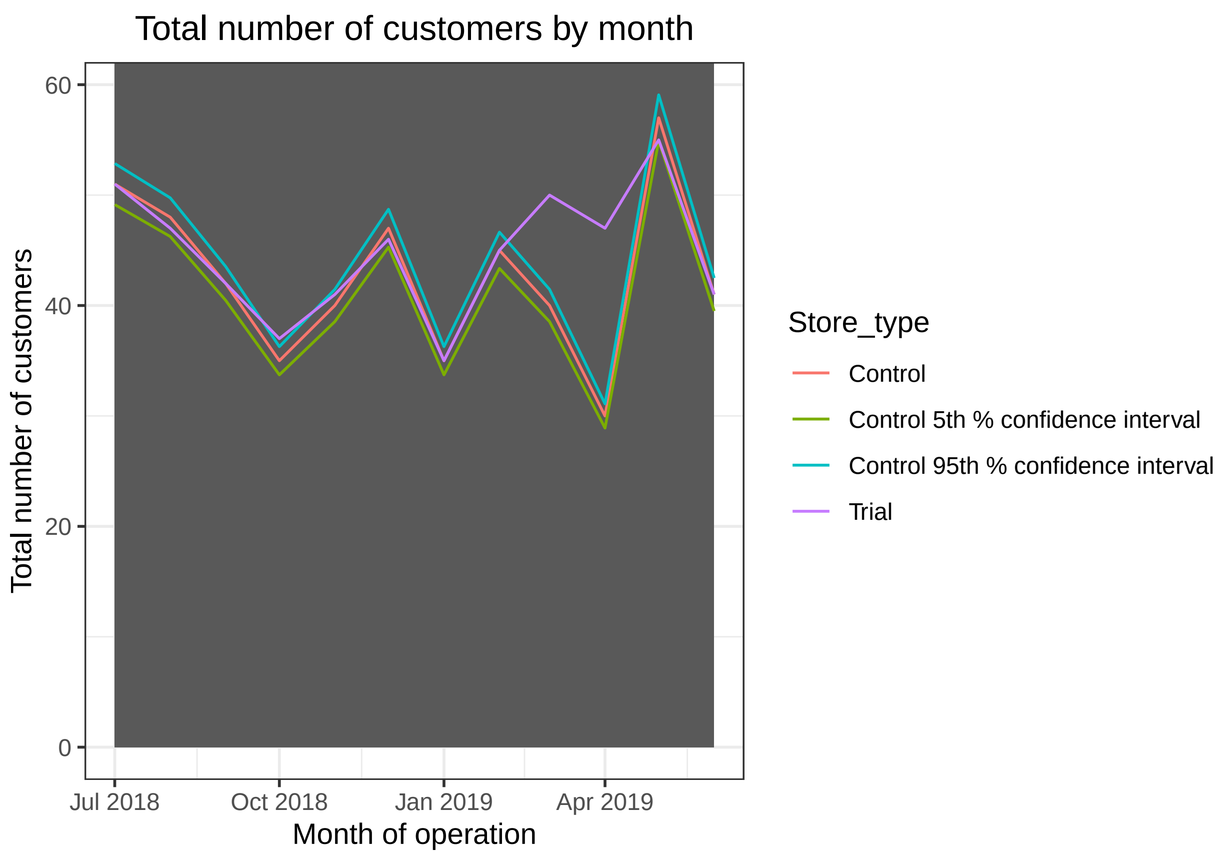
```
percentageDiff <- merge(scaledControlCustomers[, c("YEARMONTH", "controlCustomers")],
measureOverTimeCusts[STORE_NBR == trial_store,c("nCustomers", "YEARMONTH")],
by = "YEARMONTH"
)[, percentageDiff := abs(controlCustomers-nCustomers)/controlCustomers]
```

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

```
#### As our null hypothesis is that the trial period is the same as the pre-trial period, let's take th

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)
```

```
ggplot(trialAssessment, aes(TransactionMonth, nCusts, color = Store_type)) +
 geom_rect(data = trialAssessment, aes(xmin = min(TransactionMonth) , xmax = max(TransactionMonth) , ym
 color = NULL), show.legend = FALSE) +
 geom_line() +
 labs(x = "Month of operation", y = "Total number of customers", title = "Total number of customers by m
```




Great, they look visuslly similar.Now let's repeat finding the control store and assessing the impact of the trial for each of the other two trial stores.

### Trial Store 86

```
measureOverTime86 <- 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", "YEAR

measureOverTime86
```

```
##       STORE_NBR YEARMONTH totSales nCustomers nTxnPerCust nChipsPerTxn
##    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
##   ---
## 3165:       272    201902    395.5         45    1.066667     1.895833
## 3166:       272    201903    442.3         50    1.060000     1.905660
## 3167:       272    201904    445.1         54    1.018519     1.909091
## 3168:       272    201905    314.6         34    1.176471     1.775000
## 3169:       272    201906    312.1         34    1.088235     1.891892
##       avgPricePerUnit
##    1:        3.337097
##    2:        3.261111
##    3:        3.717333
##    4:        3.243103
##    5:        3.378947
##   ---
## 3165:        4.346154
## 3166:        4.379208
## 3167:        4.239048
## 3168:        4.430986
## 3169:        4.458571
```

Calculating magnitude and correlation for sales and customers

```
trial_store <- 86

corr_nSales <- calculateCorrelation(preTrialMeasures, quote(totSales),trial_store)
magnitude_nSales <- calculateMagnitudeDistance(preTrialMeasures, quote(totSales), trial_store)

corr_nCustomers <- calculateCorrelation(preTrialMeasures, quote(nCustomers), trial_store)
magnitude_nCustomers <- calculateMagnitudeDistance(preTrialMeasures, quote(nCustomers), trial_store)
```

```
corr_nSales[order(-corr_measure)]
```

```
##      Store1 Store2 corr_measure
##   1:     86     86    1.0000000
##   2:     86    155    0.8778817
##   3:     86    132    0.8465166
```

```
##   4:     86    240    0.8250658
##   5:     86    222    0.7950753
##  ---
## 256:     86    254   -0.7935056
## 257:     86     39   -0.8081209
## 258:     86    108   -0.8404129
## 259:     86    256   -0.8474008
## 260:     86    120   -0.8726932
```

```
corr_nCustomers[order(-corr_measure)]
```

```
##      Store1 Store2 corr_measure
##   1:     86     86    1.0000000
##   2:     86    155    0.9428756
##   3:     86    114    0.8553390
##   4:     86    260    0.8465020
##   5:     86    176    0.7963798
##  ---
## 256:     86    270   -0.7672673
## 257:     86     63   -0.7924024
## 258:     86    120   -0.8150968
## 259:     86    259   -0.8519630
## 260:     86     23   -0.9435589
```

```
magnitude_nSales[order(-mag_measure)]
```

```
##      Store1 Store2 mag_measure
##   1:     86     86 1.000000000
##   2:     86    155 0.962963667
##   3:     86    109 0.961984849
##   4:     86    222 0.959116232
##   5:     86    225 0.956330300
##  ---
## 256:     86    267 0.018292849
## 257:     86    198 0.017169411
## 258:     86    140 0.016637560
## 259:     86    177 0.014237070
## 260:     86     99 0.009688128
```

```
magnitude_nCustomers[order(-mag_measure)]
```

```
##      Store1 Store2 mag_measure
##   1:     86     86  1.00000000
##   2:     86    155  0.98503729
##   3:     86    225  0.96736666
##   4:     86    109  0.96593973
##   5:     86    229  0.96201740
##  ---
## 256:     86    244  0.02729919
## 257:     86    146  0.02679825
## 258:     86     99  0.02435524
## 259:     86    258  0.02203824
## 260:     86    198  0.02016350
```

```r
score_nSales <- merge(corr_nSales, magnitude_nSales, by = c("Store1", "Store2"))[ , scoreNSales := (cor
score_nSales[order(-scoreNSales)]
```

```
##      Store1 Store2 corr_measure mag_measure scoreNSales
##   1:     86     86    1.0000000  1.00000000   1.0000000
##   2:     86    155    0.8778817  0.96296367   0.9204227
##   3:     86    222    0.7950753  0.95911623   0.8770958
##   4:     86    109    0.7882995  0.96198485   0.8751422
##   5:     86    138    0.7598638  0.92371947   0.8417916
##  ---
## 256:     86     52   -0.6016292  0.03429558  -0.2836668
## 257:     86    254   -0.7935056  0.15786730  -0.3178192
## 258:     86    120   -0.8726932  0.17268762  -0.3500028
## 259:     86     42   -0.7457195  0.01979896  -0.3629603
## 260:     86    146   -0.7751274  0.01899800  -0.3780647
```

```r
score_nCustomers <- merge(corr_nCustomers, magnitude_nCustomers, by = c("Store1", "Store2"))[ , scoreNC
score_nCustomers[order(-scoreNCust)]
```

```
##      Store1 Store2 corr_measure mag_measure scoreNCust
##   1:     86     86    1.0000000  1.00000000  1.0000000
##   2:     86    155    0.9428756  0.98503729  0.9639565
##   3:     86    114    0.8553390  0.93550833  0.8954237
##   4:     86    109    0.7707780  0.96593973  0.8683589
##   5:     86    225    0.7337914  0.96736666  0.8505790
##  ---
## 256:     86    127   -0.5313244  0.04880948 -0.2412575
## 257:     86    177   -0.5724159  0.03748414 -0.2674659
## 258:     86     52   -0.5944594  0.04116568 -0.2766469
## 259:     86     42   -0.6649524  0.04027158 -0.3123404
## 260:     86    146   -0.6545983  0.02679825 -0.3139000
```

```r
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 86
control_store <- score_Control[Store1 == trial_store, ][order(-finalControlScore)][2, Store2]
control_store
```

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

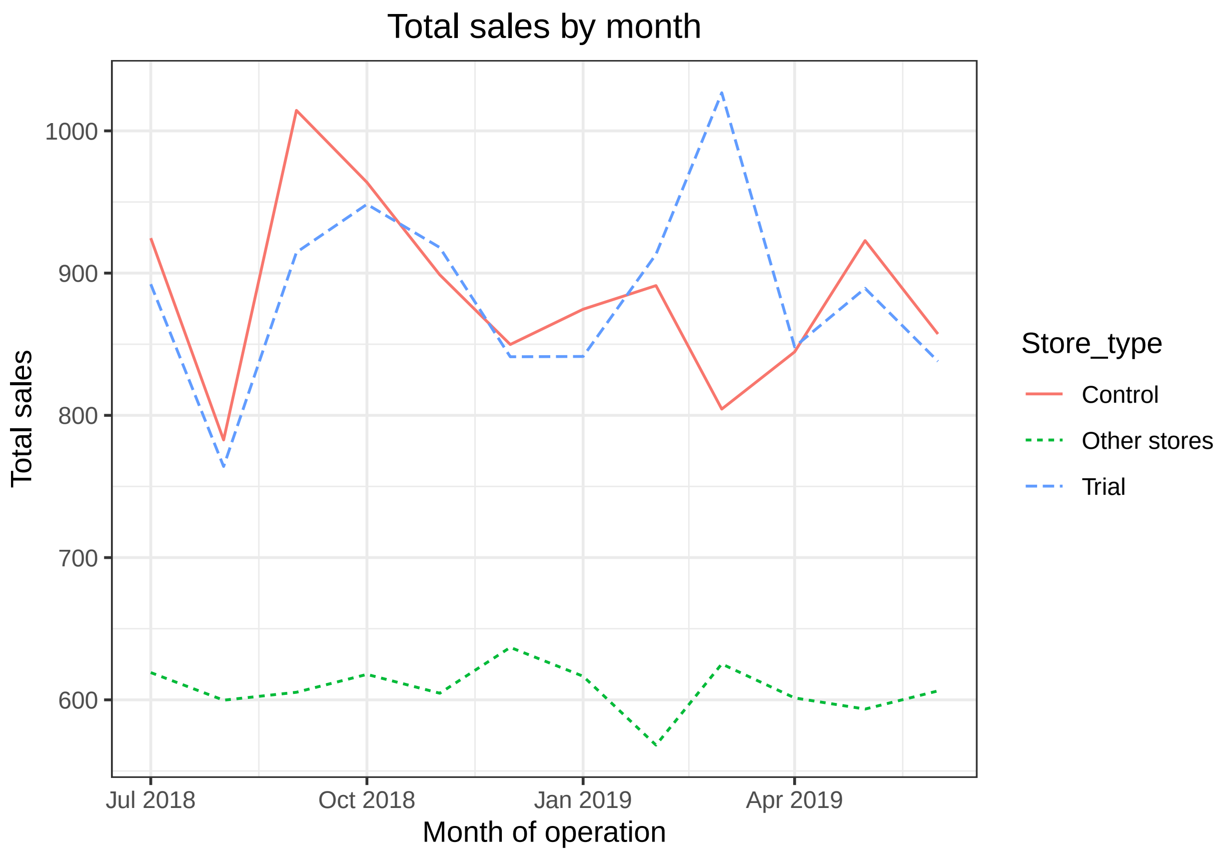
Looks like store 155 will be a control store for trial store 86. Again, let's check visually if the drivers are indeed similar in the period before the trial. We'll look at total sales first.

### Including Plots

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.

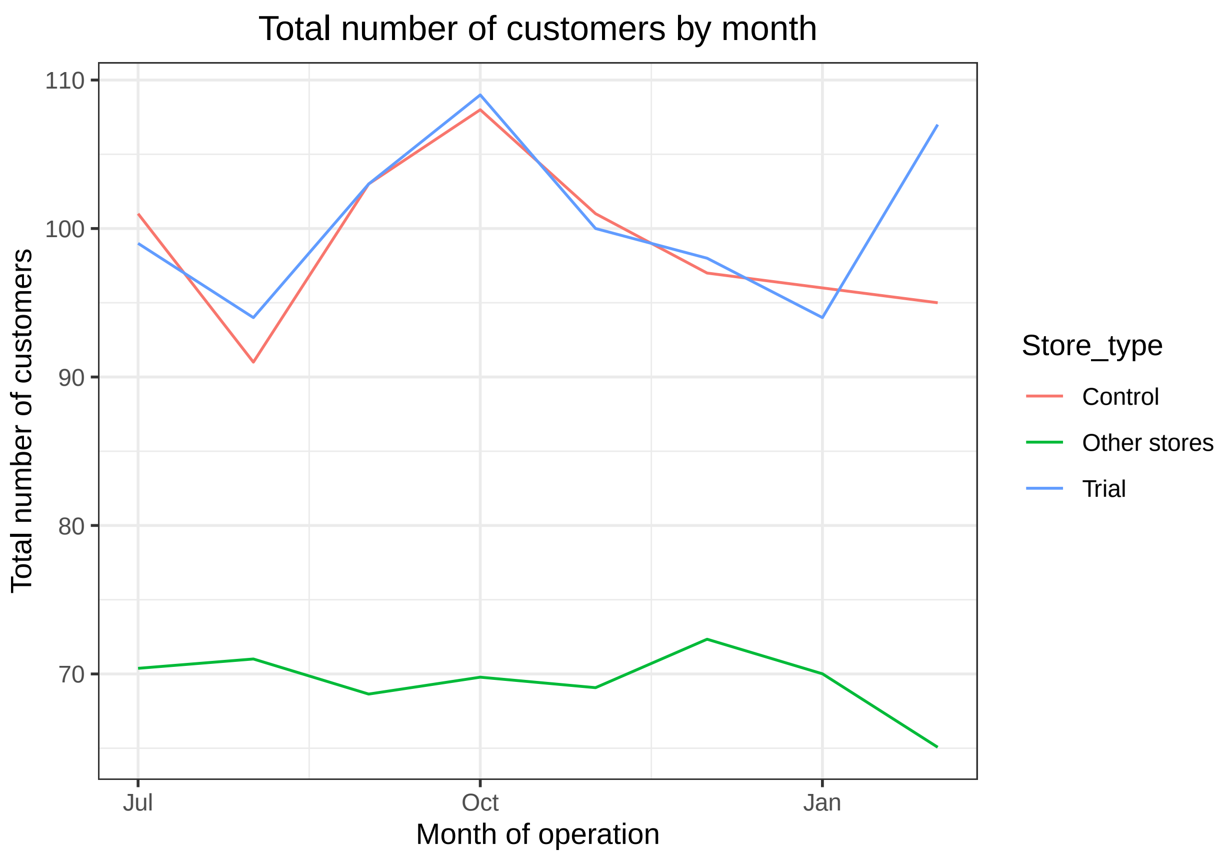
```
measureOverTimeSales <- measureOverTime86
pastSales <- measureOverTimeSales[, Store_type:= ifelse(STORE_NBR == trial_store, "Trial", ifelse(STORE_
```

```
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")
```




Great, sales are trending in a similar way. Next, number of customers.

```
measureOverTimeCusts <- measureOverTime86
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 o
```




Good, the trend in number of customers is also similar. Let's now assess the impact of the trial on sales.

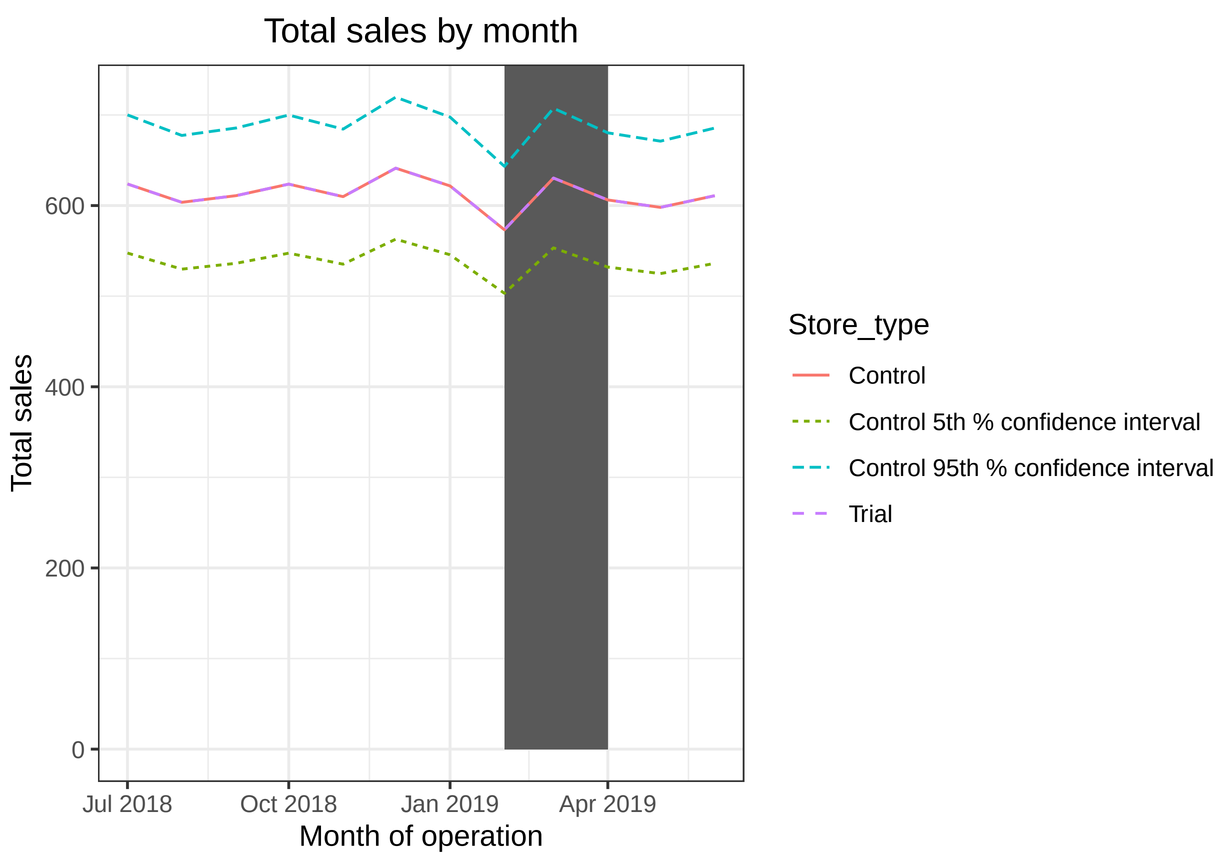
```
#### Scale pre-trial control sales to match pre-trial trial store sales
scalingFactorForControlSales <- preTrialMeasures[STORE_NBR == trial_store &
YEARMONTH < 201902, sum(totSales)]/preTrialMeasures[STORE_NBR ==
control_store & YEARMONTH < 201902, sum(totSales)]
#### Apply the scaling factor
measureOverTimeSales <- measureOverTime86
scaledControlSales <- measureOverTimeSales[STORE_NBR == control_store, ][ ,controlSales := totSales * s
#### 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]
```

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

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")
```

```r
][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)
```

```r
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")
```




The results show that the trial in store 86 is not significantly different to its control store in the trial period as the trial store performance lies inside 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 the number of customers as well.

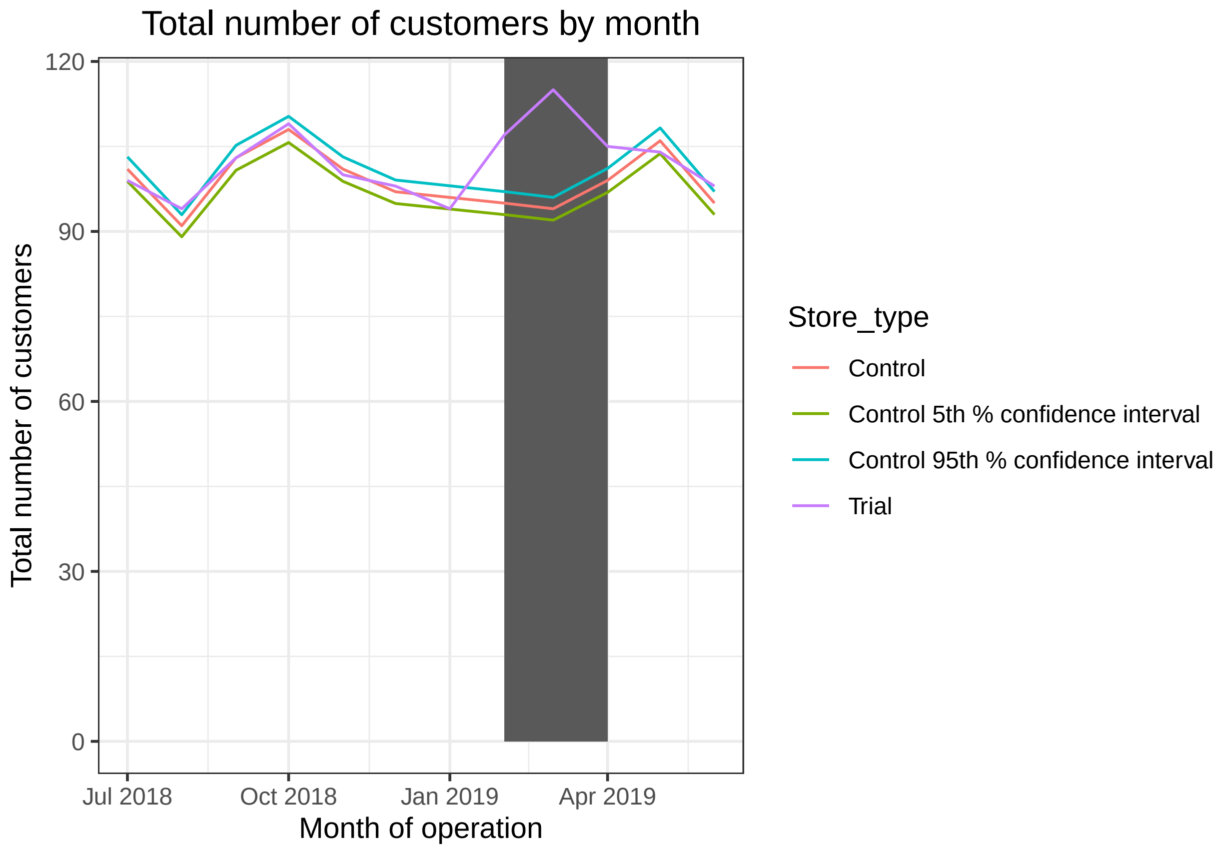
```r
#### Scale pre-trial control customers to match pre-trial trial store customers
scalingFactorForControlCust <- preTrialMeasures[STORE_NBR == trial_store &
YEARMONTH < 201902, sum(nCustomers)]/preTrialMeasures[STORE_NBR ==
control_store & YEARMONTH < 201902, sum(nCustomers)]
#### Apply the scaling factor
measureOverTimeCusts <- measureOverTime86
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"))
]
```

```r
#### Calculate the percentage difference between scaled control sales and trial sales

percentageDiff <- merge(scaledControlCustomers[, c("YEARMONTH","controlCustomers")],measureOverTime[STO
by = "YEARMONTH")[, percentageDiff := abs(controlCustomers-nCustomers)/controlCustomers]
```

```r
percentageDiff
```

```r
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)
```

```r
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 mo
```




It looks like the number of customers is significantly higher in all of the three months. This seems to suggest that the trial had a significant impact on increasing the number of customers in trial store 86 but as we saw, sales were not significantly higher. We should check with the Category Manager if there were special deals in the trial store that were may have resulted in lower prices, impacting the results.

### Trial Store 88

Calculating measures over time for store 88

```
measureOverTime88 <- 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)]
```

```
trial_store <- 88

corr_nSales <- calculateCorrelation(preTrialMeasures, quote(totSales),trial_store)
magnitude_nSales <- calculateMagnitudeDistance(preTrialMeasures, quote(totSales), trial_store)

corr_nCustomers <- calculateCorrelation(preTrialMeasures, quote(nCustomers), trial_store)
magnitude_nCustomers <- calculateMagnitudeDistance(preTrialMeasures, quote(nCustomers), trial_store)
```

```r
corr_nSales[order(-corr_measure)]
```

```
##      Store1 Store2 corr_measure
##   1:     88     88    1.0000000
##   2:     88    159    0.9031856
##   3:     88    204    0.8857742
##   4:     88    134    0.8642935
##   5:     88      1    0.8136360
##  ---
## 256:     88    272   -0.7727724
## 257:     88     23   -0.8016518
## 258:     88      8   -0.8162965
## 259:     88     48   -0.8571420
## 260:     88    230   -0.9088829
```

```r
corr_nCustomers[order(-corr_measure)]
```

```
##      Store1 Store2 corr_measure
##   1:     88     88    1.0000000
##   2:     88    237    0.9473262
##   3:     88     14    0.9429762
##   4:     88    178    0.9394660
##   5:     88     35    0.8995936
##  ---
## 256:     88     55   -0.6975325
## 257:     88    227   -0.7299425
## 258:     88    247   -0.7901029
## 259:     88    258   -0.8258499
## 260:     88    133   -0.8354265
```

```r
magnitude_nSales[order(-mag_measure)]
```

```
##      Store1 Store2 mag_measure
##   1:     88     88 1.000000000
##   2:     88    237 0.956075659
##   3:     88    203 0.950774802
##   4:     88     40 0.939013891
##   5:     88    199 0.923715270
##  ---
## 256:     88    267 0.011843977
## 257:     88    198 0.011412796
## 258:     88    140 0.010775208
## 259:     88    177 0.009174325
## 260:     88     99 0.006404420
```

```r
magnitude_nCustomers[order(-mag_measure)]
```

```
##      Store1 Store2 mag_measure
##   1:     88     88  1.00000000
##   2:     88    237  0.98758568
##   3:     88    203  0.94319890
```

```
##   4:     88     40  0.94058826
##   5:     88    165  0.93288626
##  ---
## 256:     88    244  0.02198076
## 257:     88    146  0.02072457
## 258:     88     99  0.01973725
## 259:     88    258  0.01777293
## 260:     88    198  0.01609676
```

Create a combined score composed of correlation and magnitude by merging the correlations table and the magnitudes table, for each driver.

```
score_nSales <- merge(corr_nSales, magnitude_nSales, by = c("Store1", "Store2"))[ , scoreNSales := (cor

score_nCustomers <- merge(corr_nCustomers, magnitude_nCustomers, by = c("Store1", "Store2"))[ , scoreNCu
```

Select control stores based on the highest matching store (closest to 1 but not the store itself, i.e. the second ranked highest store)

```
score_Control <- merge(score_nSales, score_nCustomers, by = c("Store1","Store2"))
score_Control[, finalControlScore := scoreNSales * 0.5 + scoreNCust * 0.5]

control_store <- score_Control[Store1 == trial_store, ][order(-finalControlScore)][2, Store2]
control_store
```

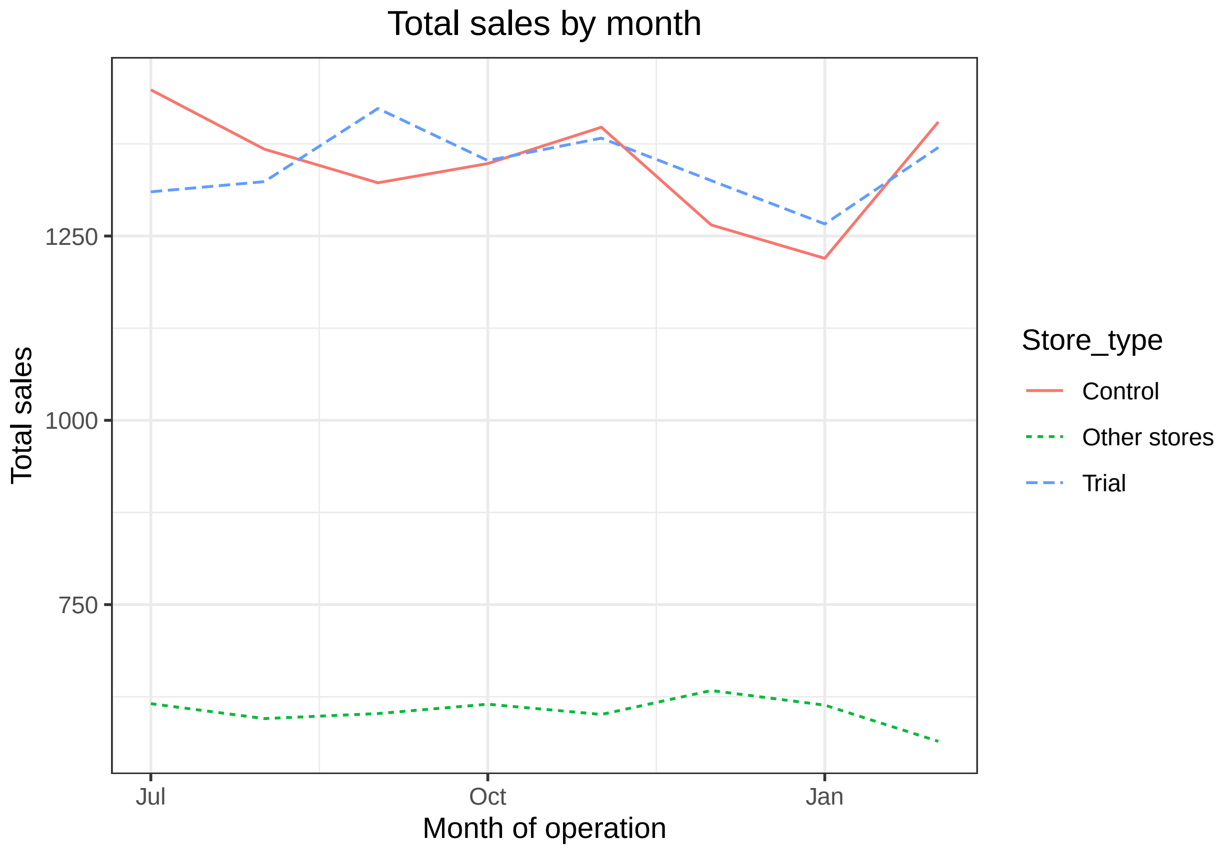
```
## [1] 237
```

We've now found store 237 to be a suitable control store for trial store 88. Again, let's check visually if the drivers are indeed similar in the period before the trial.

### Including Plots

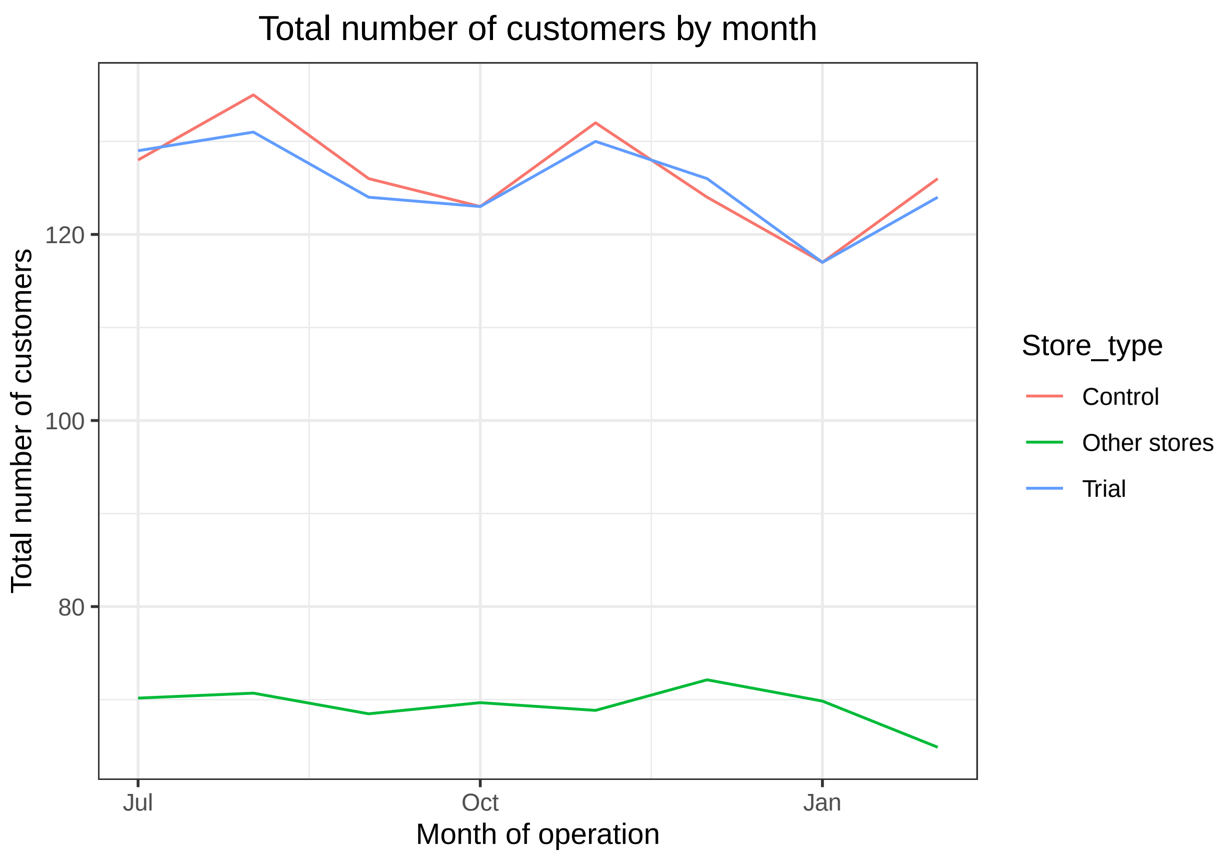
Now that we have found a control store, let's check visually if the drivers are indeed similar in the period before the trial. We'll look at total sales first.

```
measureOverTimeSales <- measureOverTime88
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 , ]

#### Plotting this in a graph
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")
```




Great, the trial and control stores have similar total sales. Next, number of customers.

```
measureOverTimeCusts <- measureOverTime88
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 , ]

#### Plotting this in a graph
ggplot(pastCustomers, aes(TransactionMonth, numberCustomers, color = Store_type)) +
  geom_line() + labs(x = "Month of operation", y = "Total number of customers", title = "Total number o
```




Total number of customers of the control and trial stores are also similar. Let's now assess the impact of the trial on sales.

```
#### 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 <- measureOverTime88
scaledControlSales <- measureOverTimeSales[STORE_NBR == control_store, ][ ,controlSales := totSales * s
```

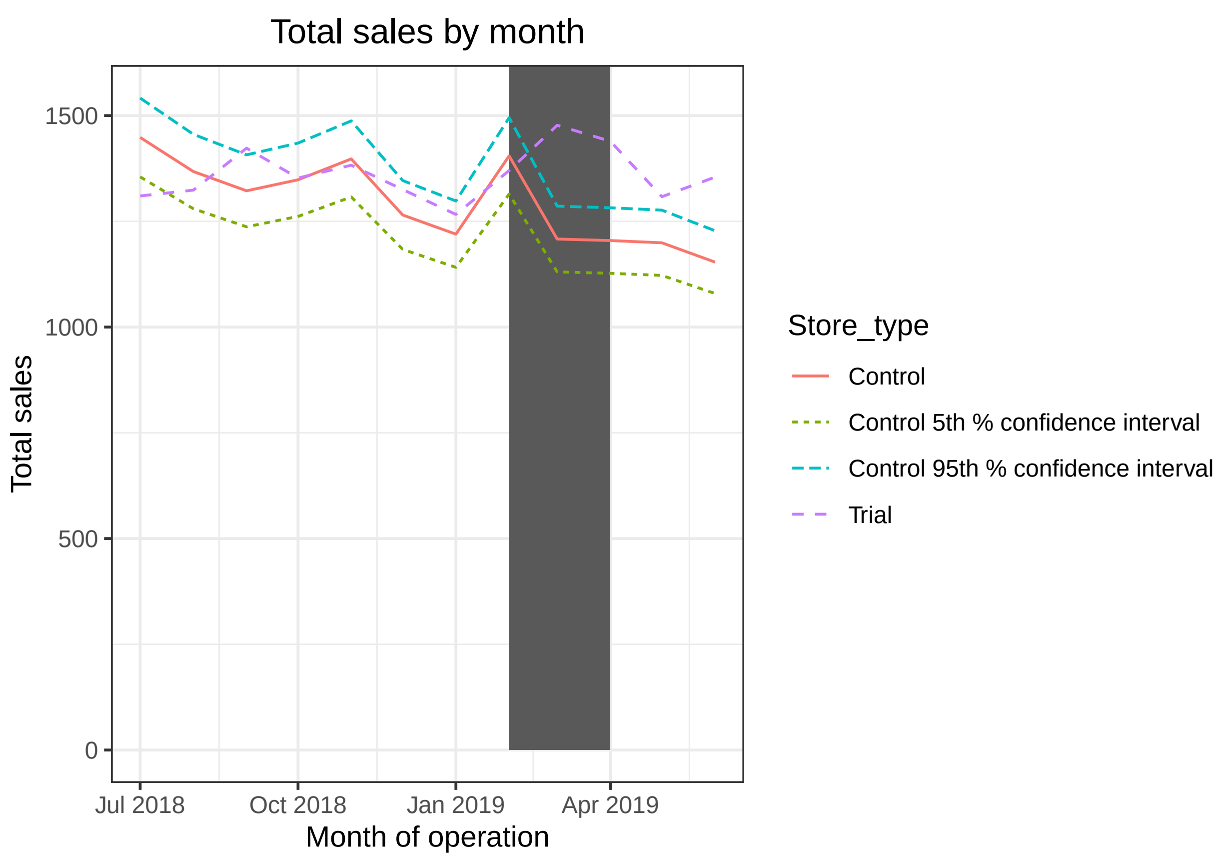
```
#### Calculate the percentage difference between scaled control sales and trial sales
percentageDiff <- merge(scaledControlSales[, c("YEARMONTH", "controlSales")],measureOverTime[STORE_NBR =
```

```
percentageDiff
```

```
stdDev <- sd(percentageDiff[YEARMONTH < 201902 , percentageDiff])
degreesOfFreedom <- 7

measureOverTimeSales <- measureOverTime88
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")
```

```r
][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)
```

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



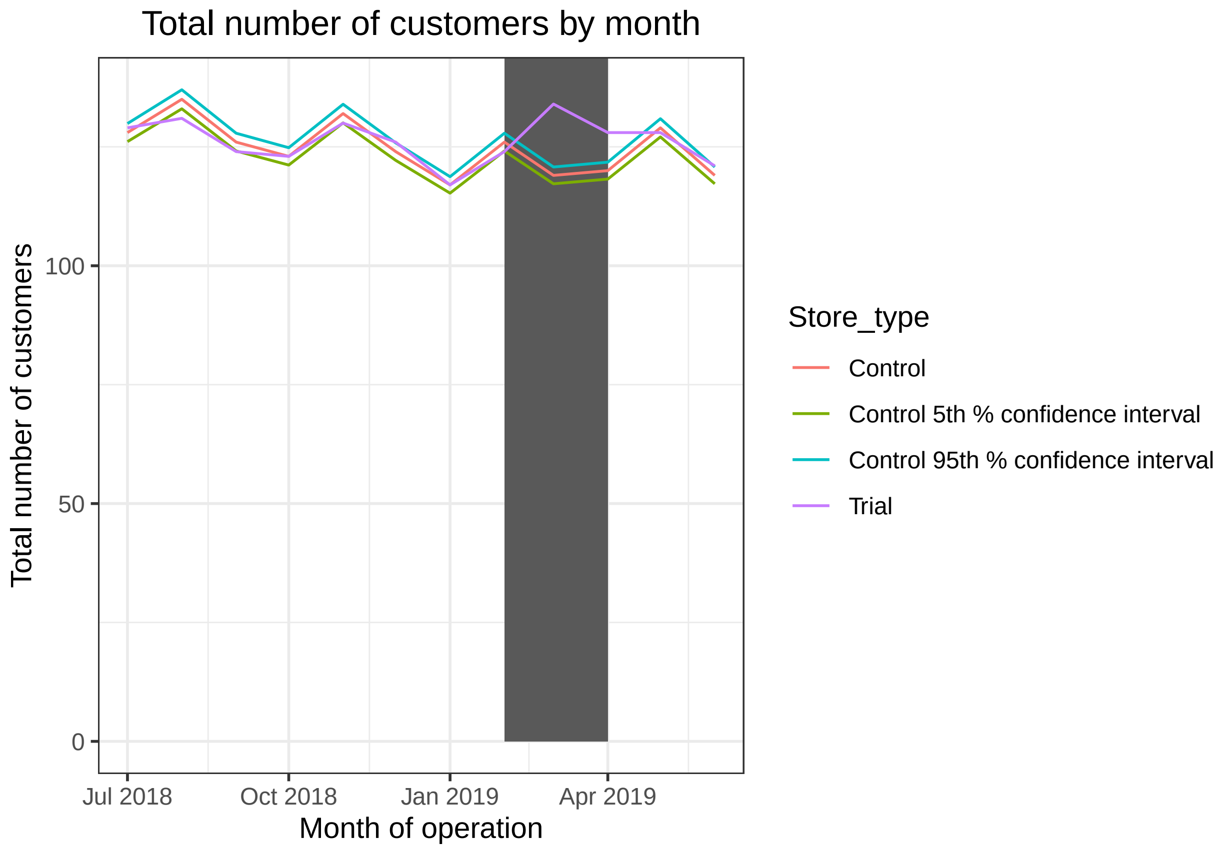

The results show that the trial in store 88 is significantly different to its control store in the trial period as the trial store performance lies outside of the 5% to 95% confidence interval of the control store in two of the three trial months. Let's have a look at assessing this for number of customers as well.

```
scalingFactorForControlCust <- preTrialMeasures[STORE_NBR == trial_store &
YEARMONTH < 201902, sum(nCustomers)]/preTrialMeasures[STORE_NBR ==
control_store & YEARMONTH < 201902, sum(nCustomers)]
#### Apply the scaling factor
measureOverTimeCusts <- measureOverTime88
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"))
]
```

```
percentageDiff <- merge(scaledControlCustomers[, c("YEARMONTH","controlCustomers")],measureOverTime[STO
by = "YEARMONTH")[, percentageDiff := abs(controlCustomers-nCustomers)/controlCustomers]
```

```
percentageDiff
```

```
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)

#### 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 m
```




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

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.

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.