Task 2

Lakmini Herath

2024-07-31

Solution template for Task 2

This file is a solution template for the Task 2 of the Quantium Virtual Internship. It will walk you through the analysis, providing the scaffolding for your solution with gaps left for you to fill in yourself.

Load required libraries and datasets

```
filePath <- "E:/DevOP/quantum/"
data <- fread(paste0(filePath,"QVI_data.csv"))

#### Set themes for plots
theme_set(theme_bw())
theme_update(plot.title = element_text(hjust = 0.5))
```

```
head(data)
```

Assign the data files to data.tables

```
##    LYLTY_CARD_NBR       DATE STORE_NBR TXN_ID PROD_NBR
##             <int>     <IDat>     <int>  <int>    <int>
## 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
## 6:           1005 2018-12-28         1      6       86
##                                   PROD_NAME PROD_QTY TOT_SALES PACK_SIZE
##                                      <char>    <int>     <num>     <int>
## 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
## 6:                     Cheetos Puffs 165g        1       2.8       165
##         BRAND              LIFESTAGE PREMIUM_CUSTOMER
##        <char>                 <char>           <char>
```

```
## 1:    Natural  YOUNG SINGLES/COUPLES          Premium
## 2:        Red  YOUNG SINGLES/COUPLES       Mainstream
## 3:      Grain         YOUNG FAMILIES           Budget
## 4:    Natural         YOUNG FAMILIES           Budget
## 5: WOOLWORTHS  OLDER SINGLES/COUPLES       Mainstream
## 6:    Cheetos MIDAGE 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.

```r
#### Calculate these measures over time for each store
#### Add a new month ID column in the data with the format yyyymm.

#monthID <- format(as.Date(data$DATE),"%Y%m")
#data[, YEARMONTH := monthID]

#data$YEARMONTH <- as.numeric(as.character(data$YEARMONTH))
data[, YEARMONTH := year(DATE)*100 + month(DATE)]
head(data)
```

```
##    LYLTY_CARD_NBR       DATE STORE_NBR TXN_ID PROD_NBR
##             <int>     <IDat>     <int>  <int>    <int>
## 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
## 6:           1005 2018-12-28         1      6       86
##                                  PROD_NAME PROD_QTY TOT_SALES PACK_SIZE
##                                     <char>    <int>     <num>     <int>
## 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
## 6:                    Cheetos Puffs 165g        1       2.8       165
##         BRAND              LIFESTAGE PREMIUM_CUSTOMER YEARMONTH
##        <char>                 <char>           <char>     <num>
## 1:    Natural  YOUNG SINGLES/COUPLES          Premium    201810
## 2:        Red  YOUNG SINGLES/COUPLES       Mainstream    201809
## 3:      Grain         YOUNG FAMILIES           Budget    201903
## 4:    Natural         YOUNG FAMILIES           Budget    201903
## 5: WOOLWORTHS  OLDER SINGLES/COUPLES       Mainstream    201811
## 6:    Cheetos MIDAGE SINGLES/COUPLES       Mainstream    201812
```

```r
#### Next, we define the measure calculations to use during the analysis.
# For each store and month calculate total sales, number of
#customers, transactions per customer, chips per customer and the average price per unit.
```

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

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
##           <int>     <num>    <num>      <int>       <num>        <num>
##    1:         1    201807    188.9         47    1.042553     1.183673
##    2:         1    201808    168.4         41    1.000000     1.268293
##    3:         1    201809    268.1         57    1.035088     1.203390
##    4:         1    201810    175.4         39    1.025641     1.275000
##    5:         1    201811    184.8         44    1.022727     1.222222
##   ---
## 3161:       272    201902    385.3         44    1.068182     1.893617
## 3162:       272    201903    421.9         48    1.062500     1.901961
## 3163:       272    201904    445.1         54    1.018519     1.909091
## 3164:       272    201905    314.6         34    1.176471     1.775000
## 3165:       272    201906    301.9         33    1.090909     1.888889
##       avgPricePerUnit
##                 <num>
##    1:        3.256897
##    2:        3.238462
##    3:        3.776056
##    4:        3.439216
##    5:        3.360000
##   ---
## 3161:        4.329213
## 3162:        4.349485
## 3163:        4.239048
## 3164:        4.430986
## 3165:        4.439706
```

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

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

Now we need to work out a way of ranking how similar each potential control store is to the trial store. We can calculate how correlated the performance of each store is to the trial store.

```r
####  Create a function to calculate correlation for a measure, looping through each control store.
#Let's define inputTable as a metric table with potential comparison stores, metric Col
#as the store metric used to calculate correlation on, and store Comparison as
#the store number of the trial store.

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

```r
                              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)
}
```

Let's write a function for this.

```r
#### Create a function to calculate a standardised magnitude distance for a measure,
#### looping through each control store calculate

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)
    }

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

  distTable <- merge(calcDistTable, minMaxDist, by = c("Store1", "YEARMONTH"))

  distTable[, magnitudeMeasure := 1 - (measure - minDist)/(maxDist - minDist)]

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

  return(finalDistTable)
  }
```

Store 77

```r
####Use the function you created to 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
```

```
##      Store1 Store2 corr_measure
##       <num>  <num>        <num>
##   1:     77      1 -0.005382429
##   2:     77      2 -0.251182809
##   3:     77      3  0.660446832
##   4:     77      4 -0.347846468
##   5:     77      5 -0.139047983
##  ---
## 255:     77    268  0.395460337
## 256:     77    269 -0.466370424
## 257:     77    270  0.274854303
## 258:     77    271  0.195189898
## 259:     77    272 -0.179646952
```

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

corr_nCustomers
```

```
##      Store1 Store2 corr_measure
##       <num>  <num>        <num>
##   1:     77      1  0.337865596
##   2:     77      2 -0.596491730
##   3:     77      3  0.755248715
##   4:     77      4 -0.305411652
##   5:     77      5  0.224768439
##  ---
## 255:     77    268  0.369735946
## 256:     77    269 -0.247580595
## 257:     77    270 -0.009181744
## 258:     77    271  0.023634941
## 259:     77    272  0.068677178
```

```r
#### Then, 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,
#by first merging the correlations table with the magnitude table.

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

```r
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)]
```

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

```r
#### Combine scores across the drivers by first merging our sales scores and customer
#scores into a single table

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

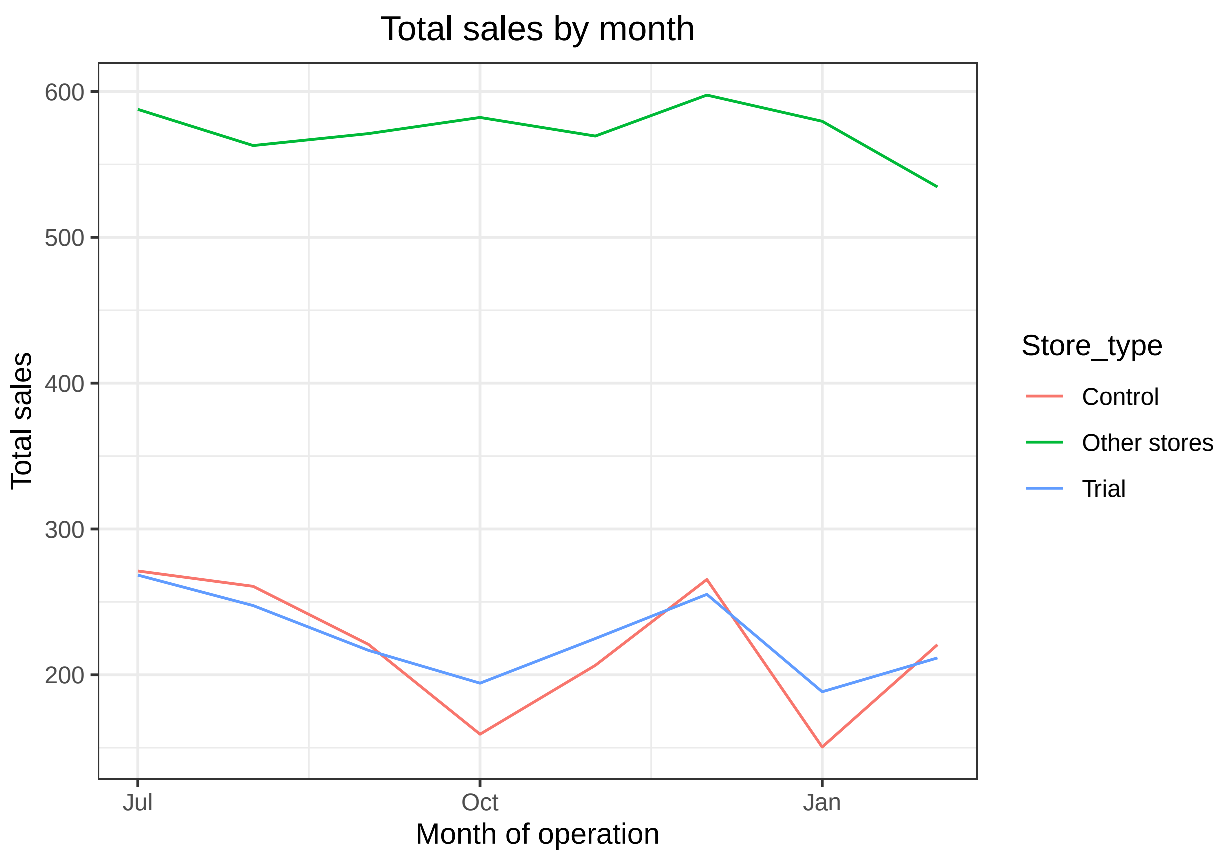
The store with the highest score is then selected as the control store since it is most similar to the trial store.

```r
#### Select control stores based on the highest matching store (closest to 1 but
#### not the store itself, i.e. the second ranked highest store)
#### Select the most appropriate control store for trial store 77 by finding the
#store with the highest final score.

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

score_Control[order(-finalControlScore)][2, Store2]
```

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

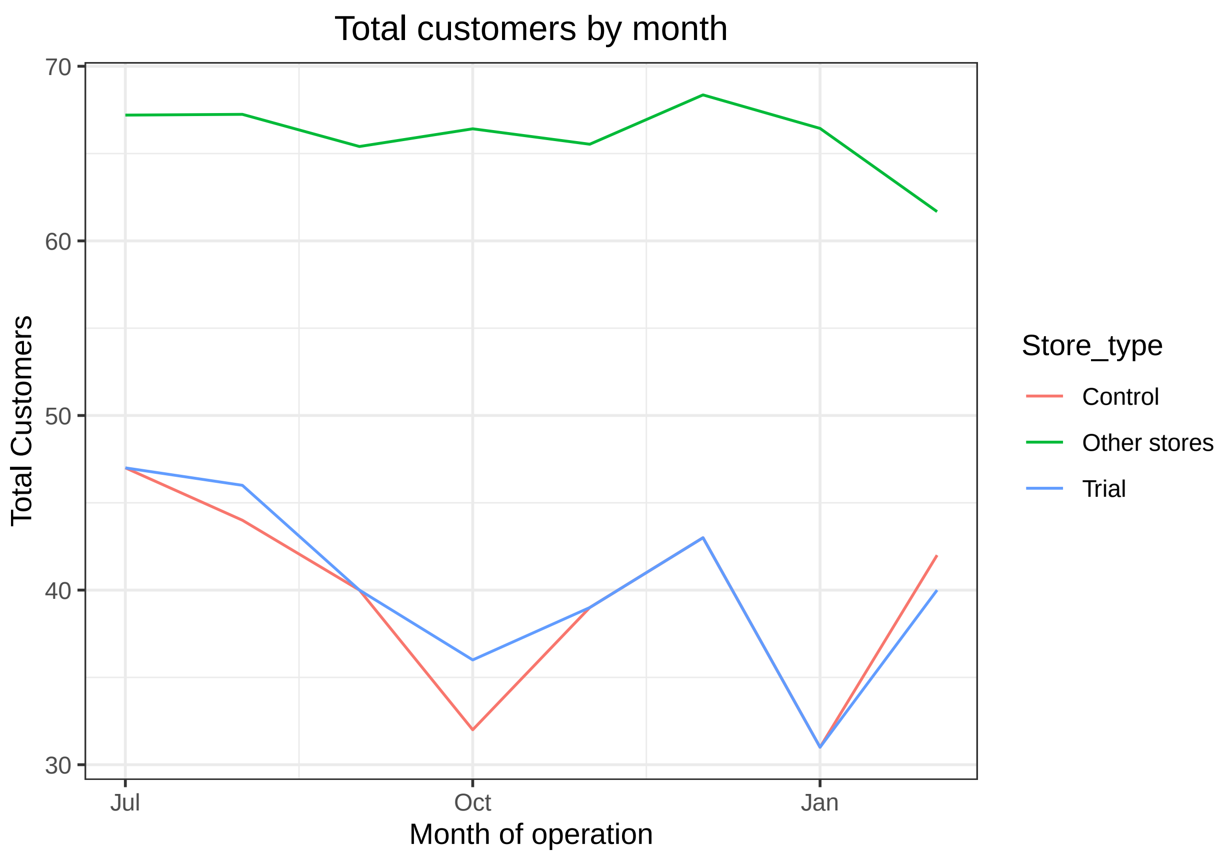
Now that we have found a control store, let's check visually if the drivers are indeed similar in the period before the trial. We'll look at total sales first.

```r
#### Visual checks on trends based on the drivers

measureOverTimeSales <- measureOverTime
pastSales <- measureOverTimeSales[,
            Store_type := ifelse(STORE_NBR == trial_store, "Trial",

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.

```
####Conduct visual checks on customer count trends by comparing the trial store
#to the control store and other stores.
#### Hint: Look at the previous plot.

measureOverTimeCusts <- measureOverTime
pastCustomers <- measureOverTimeCusts[,
                Store_type := ifelse(STORE_NBR == trial_store, "Trial",

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




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

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

#### Apply the scaling factor

measureOverTimeSales <- measureOverTime
scaledControlSales <- measureOverTimeSales[STORE_NBR == control_store, ][
  , controlSales := totSales * scalingFactorForControlSales]
```

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

```r
#### 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]
```

Let's see if the difference is significant!

```r
#### 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])

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

degreesOfFreedom <- 7

#### We will test with a null hypothesis of there being 0 difference between trial and control stores.
#### Calculate the t-values for the trial months. After that, find the 95th percentile of #### the t di
#### to check whether the hypothesis is statistically significant.
#### Hint: The test statistic here is (x - u)/standard deviation

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
##              <Date>     <num>
## 1:       2019-02-01  1.223912
## 2:       2019-03-01  5.633494
## 3:       2019-04-01 11.336505
```

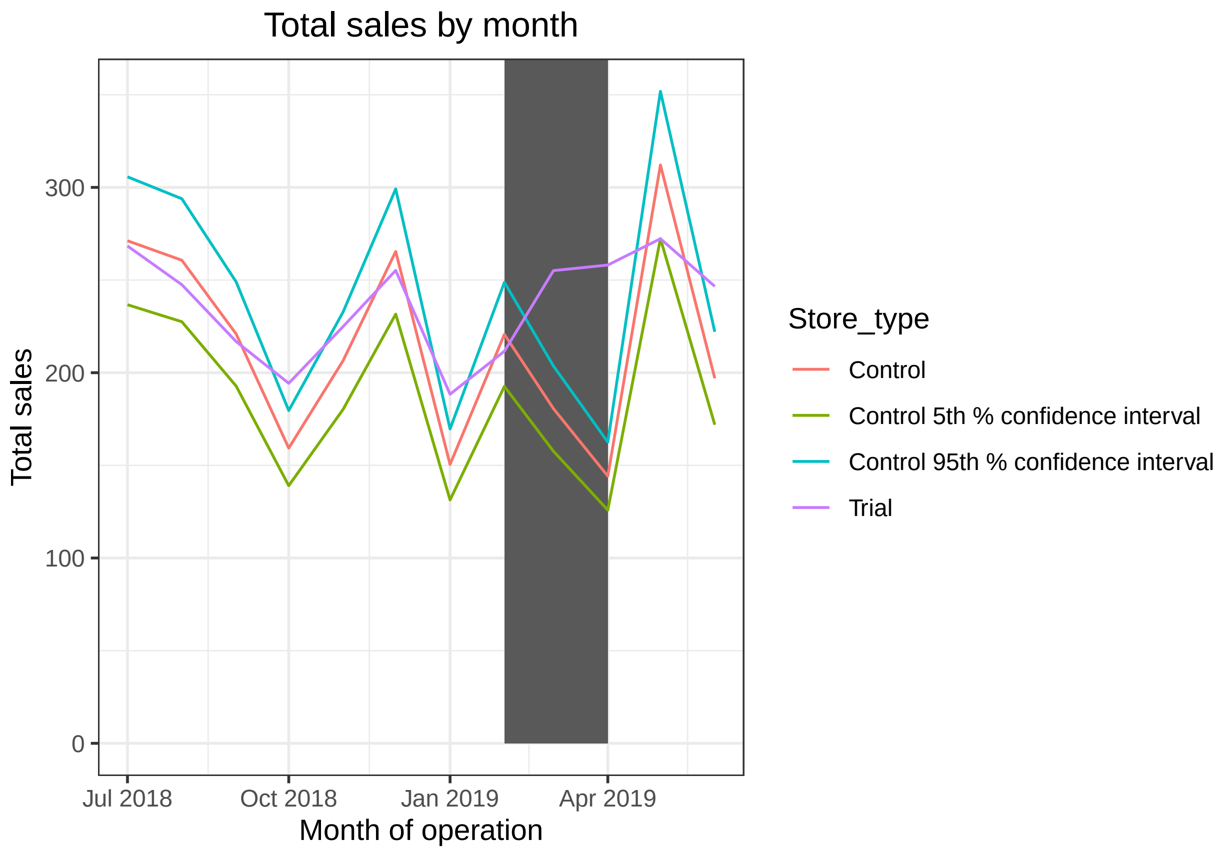
```r
qt(0.95, df = degreesOfFreedom) # 1.89
```

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

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.

```r
measureOverTimeSales <- measureOverTime

#### Trial and control store total sales
#### Create new variables Store_type, totSales and Transaction Month in the data table.

pastSales <- measureOverTimeSales[, Store_type := ifelse(STORE_NBR == trial_store, "Trial"
     ,ifelse(STORE_NBR == control_store, "Control", "Other stores"))
     ][, totSales := mean(totSales), by = 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"]

trialAssessment <- rbind(pastSales, pastSales_Controls95, pastSales_Controls5)

write_xlsx(trialAssessment, "store1_sales.xlsx")
#### Plotting these in one nice graph

ggplot(trialAssessment, aes(TransactionMonth, totSales, color = Store_type)) + geom_rect(data = trialAs
aes(xmin = min(TransactionMonth), xmax = max(TransactionMonth),  ymin = 0 , ymax = Inf, color = NULL), 
```



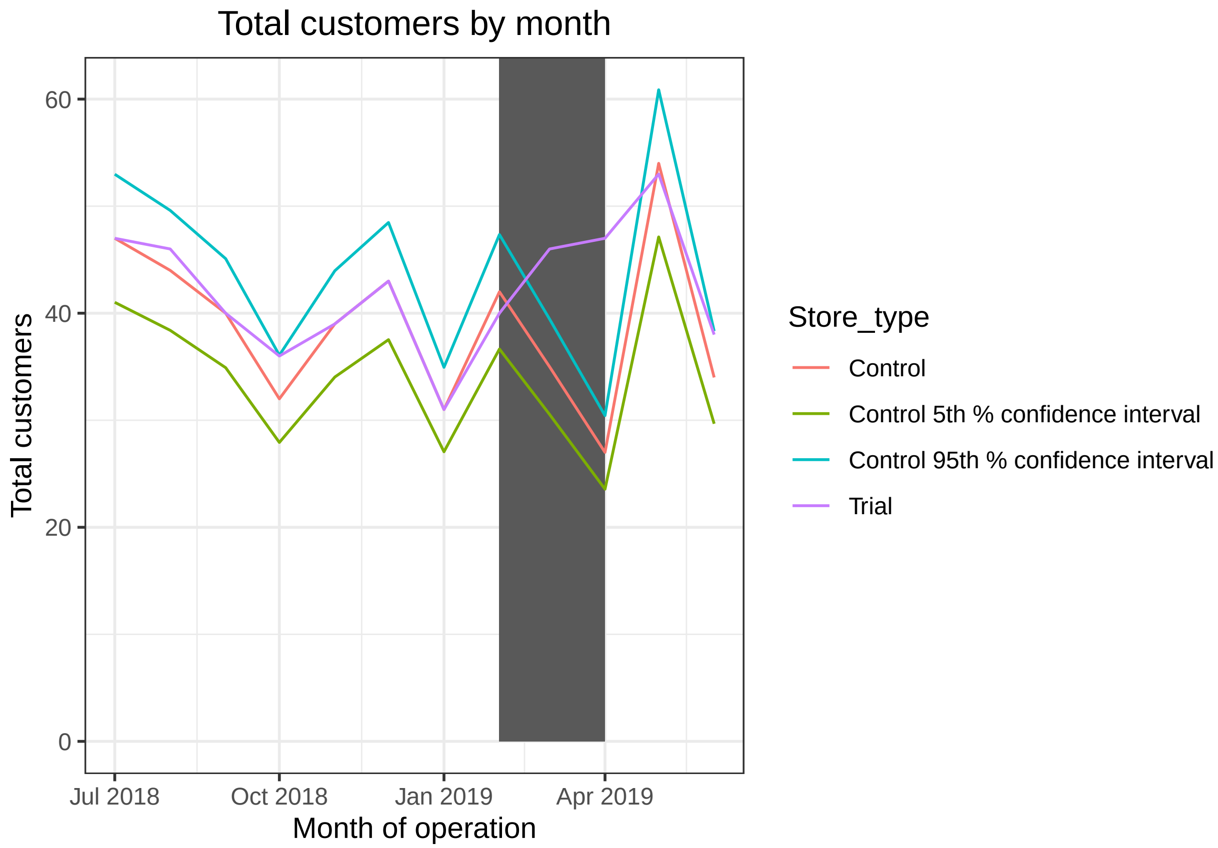
The results show that the trial in store 77 is significantly different to its control store in the trial period as the trial store performance lies outside the 5% to 95% confidence interval of the control store in two of the three trial months. Let's have a look at assessing this for number of customers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

```
#### This would be a repeat of the steps before for total sales
#### Scale pre-trial control customers to match pre-trial trial store customers
####Compute a scaling factor to align control store customer counts to our trial store.

scalingFactoForControlCust <- preTrialMeasures[STORE_NBR == trial_store&
  YEARMONTH < 201902, sum(nCustomers)]/preTrialMeasures[
  STORE_NBR == control_store & YEARMONTH < 201902, sum(nCustomers)]

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

measureOverTimeCusts <- measureOverTime
scaledControlCustomers <- measureOverTimeCusts[STORE_NBR == control_store,
```

```r
         ][, controlCustomers := nCustomers*scalingFactoForControlCust
         ][, Store_type := ifelse(STORE_NBR == trial_store, "Trial",
         ifelse(STORE_NBR == control_store,"Control", "Other stores"))]

####  calculate the percentage difference between scaled control store customers and trial customers.

percentageDiffCust <- 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!

```r
#### As our null hypothesis is that the trial period is the same as the pre-trial period,
####let's take the standard deviation based on the scaled percentage difference in the pre-trial period

stdDevc <- 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 + stdDevc * 2) ][,
                Store_type := "Control 95th % confidence interval"]

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

trialAssessmentCus <- rbind(pastCustomers, pastCustomers_Controls95, pastCustomers_Controls5)

write_xlsx(trialAssessmentCus, "store1_cus.xlsx")

####  Plot everything into one nice graph.
#### Hint: geom_rect creates a rectangle in the plot.
#### Use this to highlight the trial period in our graph.

ggplot(trialAssessmentCus, aes(TransactionMonth,  nCusts,
color = Store_type)) + geom_rect(data = trialAssessmentCus[ 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 customers", title = "Total customers by month")
```



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

#Store 86

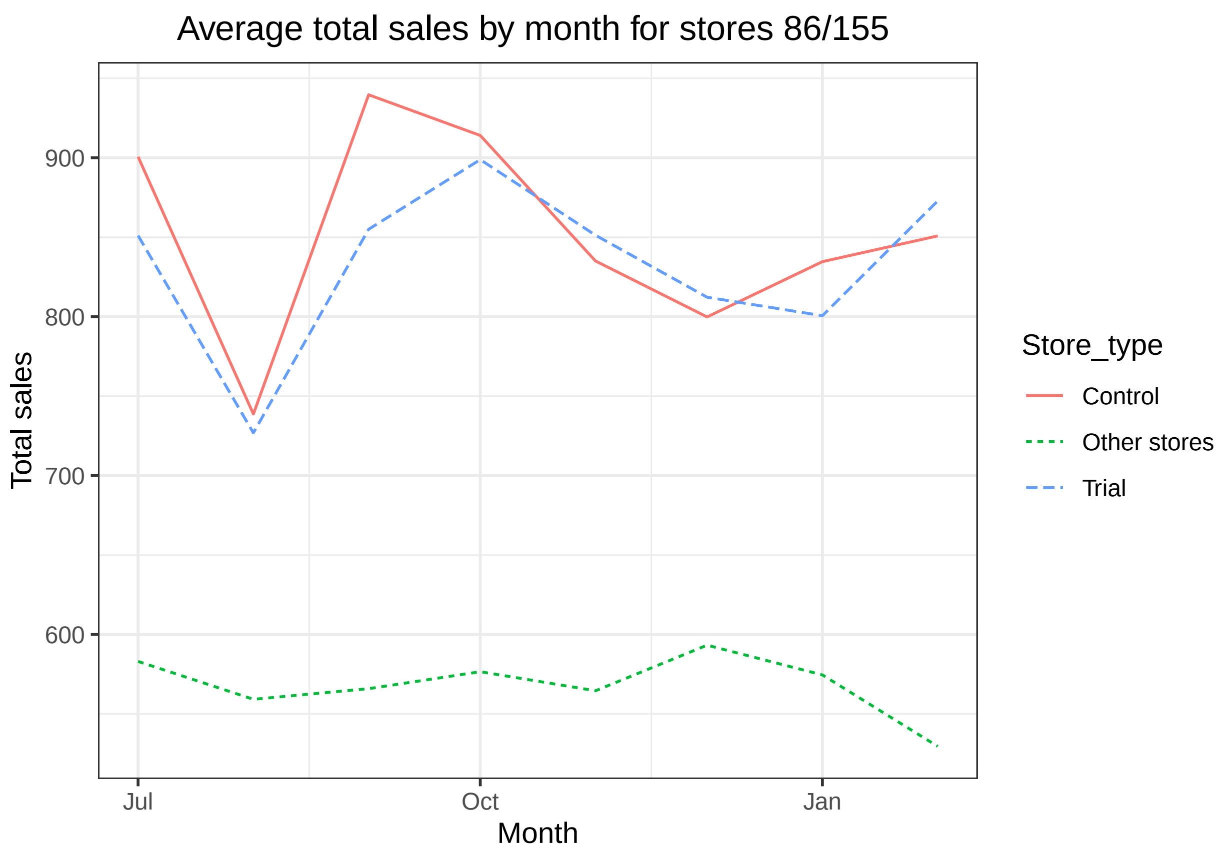
```
## Trial Store 86
#### Calculate the metrics below as we did for the first trial store.
measureOverTime <- data[, .(totSales = sum(TOT_SALES),
                            nCustomers = uniqueN(LYLTY_CARD_NBR),
                            nTxnPerCust = uniqueN(TXN_ID)/uniqueN(LYLTY_CARD_NBR),
                            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 we created earlier to calculate correlations and magnitude for each potential co
trial_store86 <- 86

corr_nSales86 <- calculateCorrelation(preTrialMeasures, quote(totSales), trial_store86)

corr_nCustomers86 <- calculateCorrelation(preTrialMeasures, quote(nCustomers), trial_store86)

#### Then, use the functions for calculating magnitude.

magnitude_nSales86 <- calculateMagnitudeDistance(preTrialMeasures, quote(totSales),trial_store86)

magnitude_nCustomers86 <- calculateMagnitudeDistance(preTrialMeasures, quote(nCustomers),trial_store86)
```

```r
#### Now, create a combined score composed of correlation and magnitude
corr_weight <- 0.5
score_nSales86 <- merge(corr_nSales86,magnitude_nSales86 ,
by = c("Store1", "Store2"))[,
scoreNSales := corr_measure * corr_weight +mag_measure * (1 - corr_weight)]

score_nCustomers86 <- merge(corr_nCustomers86,magnitude_nCustomers86,
by =c("Store1", "Store2") )[,
scoreNCust := corr_measure * corr_weight +mag_measure * (1 - corr_weight)]

#### 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

#### Combine scores across

score_Control86 <- merge(score_nSales86,score_nCustomers86 , by = c("Store1", "Store2"))
score_Control86[, finalControlScore := scoreNSales * 0.5 + scoreNCust * 0.5]

control_store86<- score_Control86[Store1 == trial_store86][order(-finalControlScore)][2, Store2]
score_Control86[order(-finalControlScore)][2, Store2]
```

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

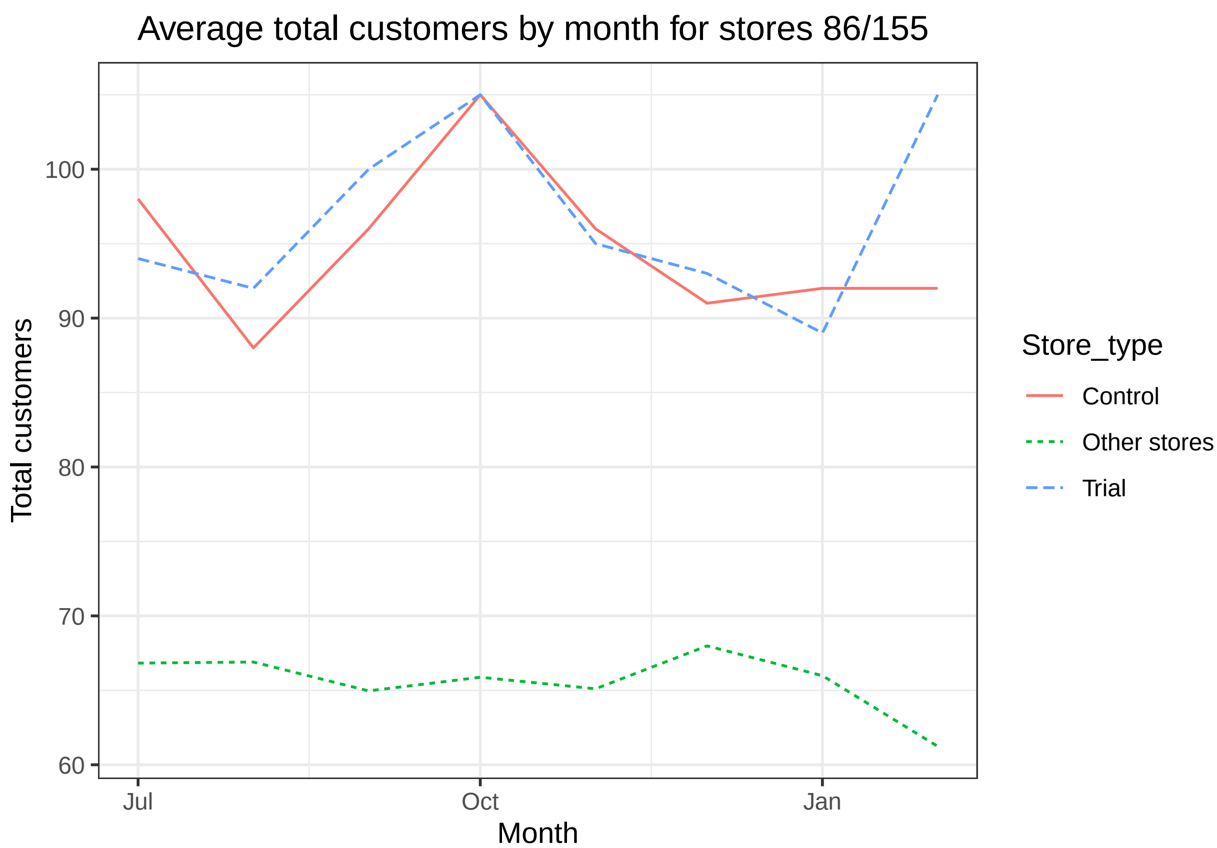
Looks like store 155 will be a control store for trial store 86. Again, let's check visually if the drivers are indeed similar in the period before the trial. We'll look at total sales first.

```r
#### Conduct visual checks on trends based on the drivers
measureOverTimeSales <- measureOverTime

pastSales86 <- measureOverTimeSales[,
              Store_type := ifelse(STORE_NBR == trial_store86, "Trial",
              ifelse(STORE_NBR == control_store86, "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(pastSales86, aes(TransactionMonth, totSales, color = Store_type)) +
  geom_line(aes(linetype = Store_type)) +
  labs(x = "Month", y = "Total sales",
       title = "Average total sales by month for stores 86/155 ")
```



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

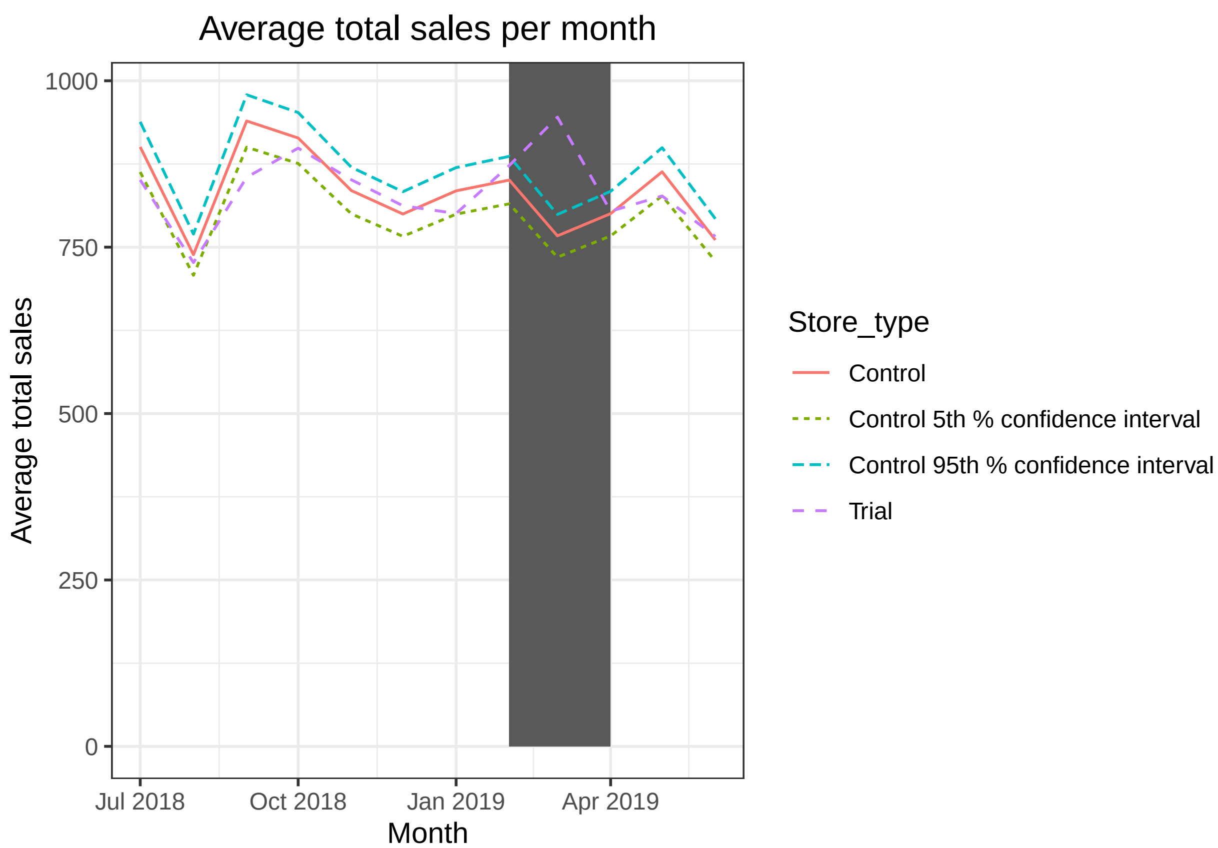
```
####  Conduct visual checks on trends based on the drivers
measureOverTimeCusts <- measureOverTime

pastCustomers86 <- measureOverTimeCusts[,
                  Store_type := ifelse(STORE_NBR == trial_store86, "Trial",
                  ifelse(STORE_NBR == control_store86, "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(pastCustomers86, aes(TransactionMonth, numberCustomers, color = Store_type)) +
  geom_line(aes(linetype = Store_type)) +
  labs(x = "Month", y = "Total customers",
       title = "Average total customers by month for stores 86/155 ")
```



Let's now assess the impact of the trial on sales.

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

scalingFactorForControlSales86 <- preTrialMeasures[STORE_NBR == trial_store86 & YEARMONTH < 201902,
sum(totSales)]/preTrialMeasures[STORE_NBR == control_store86 & YEARMONTH < 201902, sum(totSales)]

#### Apply the scaling factor
measureOverTimeSales <- measureOverTime

scaledControlSales86 <- measureOverTimeSales[STORE_NBR == control_store86,
                        ][, controlSales := totSales *scalingFactorForControlSales86]

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

#### Hint: When calculating percentage difference, remember to use absolute difference

percentageDiff86 <- merge(scaledControlSales86[, c("YEARMONTH", "controlSales")],
                  measureOverTime[STORE_NBR == trial_store86,
                  c("YEARMONTH", "totSales")], 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 th

#### Calculate the standard deviation of percentage differences during the pre-trial period
stdDev86 <- sd(percentageDiff86[YEARMONTH < 201902 , percentageDiff])
```

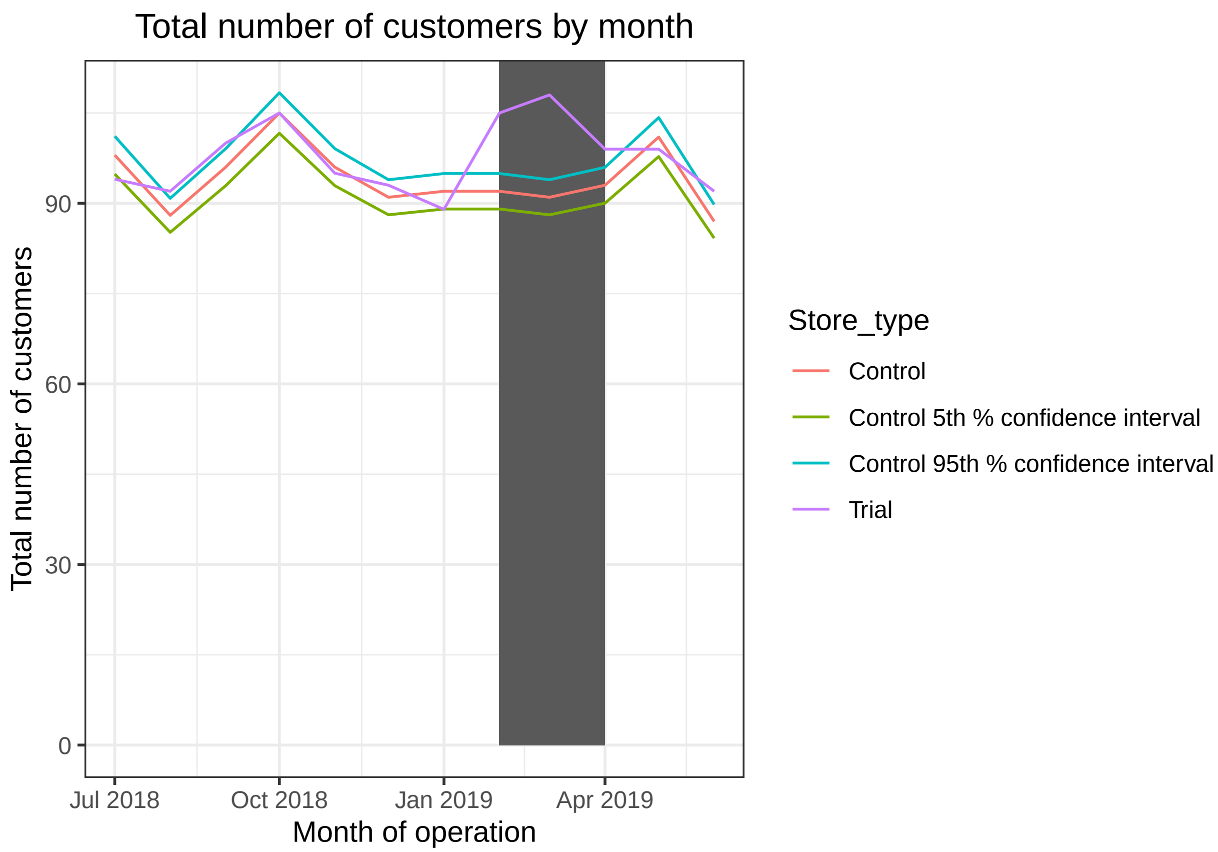
```r
degreesOfFreedom <- 7

#### Trial and control store total sales
measureOverTimeSales <- measureOverTime

pastSales86 <- measureOverTimeSales[,
              tore_type := ifelse(STORE_NBR == trial_store86, "Trial",
              ifelse(STORE_NBR == control_store86, "Control", "Other stores"))][,
              totSales := mean(totSales), by = c("YEARMONTH", "Store_type")][,
              TransactionMonth := as.Date(paste(YEARMONTH %/% 100,
                                                YEARMONTH %% 100, 1, sep = "-"),
                                        Store_type %in% c("Trial", "Control"), ]

#### Calculate the 5th and 95th percentile for control store sales.
#### Hint: The 5th and 95th percentiles can be approximated by using two standard deviations away from t

pastSales86_Controls95 <- pastSales86[Store_type == "Control",][,
                        totSales := totSales * (1 + stdDev86 * 2)][,
                        Store_type := "Control 95th % confidence interval"]

pastSales86_Controls5 <- pastSales86[Store_type == "Control",][,
                        totSales := totSales * (1 - stdDev86 * 2)][,
                        Store_type := "Control 5th % confidence interval"]

#### Hint2: Recall that the variable stdDev earlier calculates standard deviation in percentages, and no

#### Then, create a combined table with columns from pastSales, pastSales_Controls95 and pastSales_Contr

trialAssessmentSales86 <- rbind(pastSales86, pastSales86_Controls5, pastSales86_Controls95)
#### Plotting these in one nice graph

ggplot(trialAssessmentSales86, aes(TransactionMonth, totSales, color = Store_type)) +
  geom_rect(data = trialAssessmentSales86[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", y = "Average total sales", title = "Average total sales per 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.

```
#### This would be a repeat of the steps before for total sales
#### Scale pre-trial control customers to match pre-trial trial store customers

scalingFactorForControlCust86 <- preTrialMeasures[STORE_NBR == trial_store86 & YEARMONTH < 201902,
sum(nCustomers)]/preTrialMeasures[STORE_NBR == control_store86 & YEARMONTH < 201902, sum(nCustomers)]

#### Apply the scaling factor measureOverTimeCusts <- measureOverTime

scaledControlCustomers86 <- measureOverTimeCusts[STORE_NBR == control_store86,

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

percentageDiffcus86 <- merge(scaledControlCustomers86[, c("YEARMONTH","controlCustomers")],

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

stdDevcus86 <- sd(percentageDiffcus86[YEARMONTH < 201902 , percentageDiff])

degreesOfFreedom <- 7

#### Trial and control store number of customers
```

```
pastCustomers86 <- measureOverTimeCusts[, nCusts := mean(nCustomers),
                   by = c("YEARMONTH", "Store_type")][Store_type %in% c("Trial", "Control"), ]

#### Control store 95th percentile

pastCustomers86_Controls95 <- pastCustomers86[Store_type == "Control",][,
                              nCusts := nCusts * (1 + stdDevcus86 * 2)][,
                              Store_type := "Control 95th % confidence interval"]

#### Control store 5th percentile
pastCustomers86_Controls5 <- pastCustomers86[Store_type == "Control",][,
                            nCusts := nCusts * (1 - stdDevcus86 * 2)][,
                            Store_type := "Control 5th % confidence interval"]

trialAssessmentcus86 <- rbind(pastCustomers86, pastCustomers86_Controls95, pastCustomers86_Controls5)

#### Plotting these in one nice graph

ggplot(trialAssessmentcus86, aes(TransactionMonth, nCusts, color = Store_type)) +  geom_rect(data = tria
aes(xmin = min(TransactionMonth), xmax = max(TransactionMonth),  ymin = 0 , ymax = Inf, color = NULL), 
geom_line() +  labs(x = "Month of operation", y = "Total number of customers", title = "Total number of
```



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.

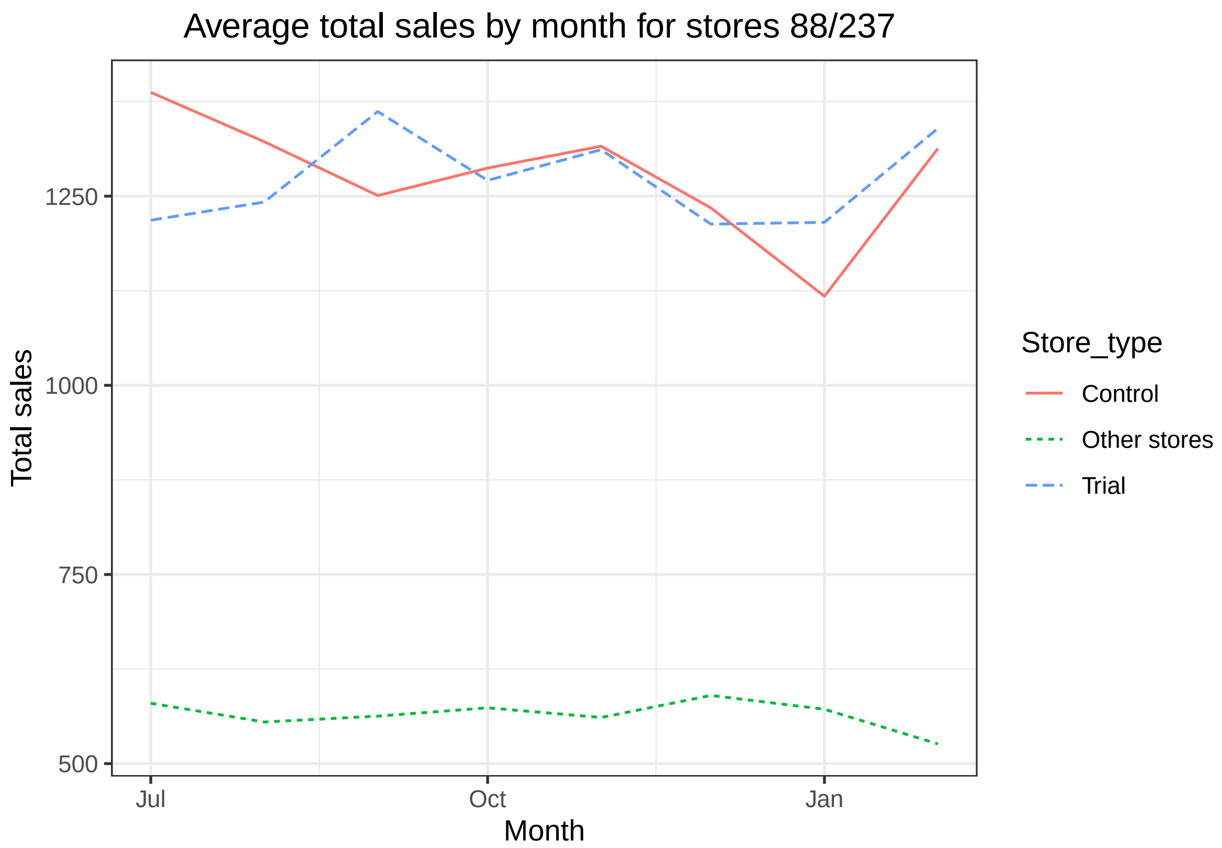
Trial store 88

```
#### Calculate the metrics below as we did for the first trial store.
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 we created earlier to calculate correlations and magnitude for each potential co
trial_store88 <- 88

corr_nSales88 <- calculateCorrelation(preTrialMeasures, quote(totSales),trial_store88)

corr_nCustomers88 <- calculateCorrelation(preTrialMeasures,
                    quote(nCustomers), trial_store88)

#### Then, use the functions for calculating magnitude.

magnitude_nSales88 <- calculateMagnitudeDistance(preTrialMeasures, quote(totSales),trial_store88)

magnitude_nCustomers88 <- calculateMagnitudeDistance(preTrialMeasures, quote(nCustomers),trial_store88)

#### Now, create a combined score composed of correlation and magnitude
corr_weight <- 0.5
score_nSales88 <- merge(corr_nSales88,magnitude_nSales88 ,
by = c("Store1", "Store2"))[,
scoreNSales := corr_measure * corr_weight +mag_measure * (1 - corr_weight)]

score_nCustomers88 <- merge(corr_nCustomers88,magnitude_nCustomers88,
by =c("Store1", "Store2") )[,
scoreNCust := corr_measure * corr_weight +mag_measure * (1 - corr_weight)]

#### 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

#### Combine scores across

score_Control88 <- merge(score_nSales88,score_nCustomers88 , by = c("Store1", "Store2"))
score_Control88[, finalControlScore := scoreNSales * 0.5 + scoreNCust * 0.5]

control_store88<- score_Control88[Store1 == trial_store88][
                  order(-finalControlScore)][2, Store2]
score_Control88[order(-finalControlScore)][2, Store2]
```

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

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.

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

pastSales88 <- measureOverTimeSales[,
              Store_type := ifelse(STORE_NBR == 88, "Trial",
              ifelse(STORE_NBR == control_store88, "Control", "Other stores"))][,
               totSales := mean(totSales), by = c("YEARMONTH", "Store_type")][,
              TransactionMonth := as.Date(paste(YEARMONTH %/% 100,
                                                YEARMONTH %% 100, 1, sep = "-"),         "%Y-%m-%d")][YEA

ggplot(pastSales88, aes(TransactionMonth, totSales, color = Store_type)) +
  geom_line(aes(linetype = Store_type)) +labs(x = "Month", y = "Total sales",
       title = "Average total sales by month for stores 88/237 ")
```



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

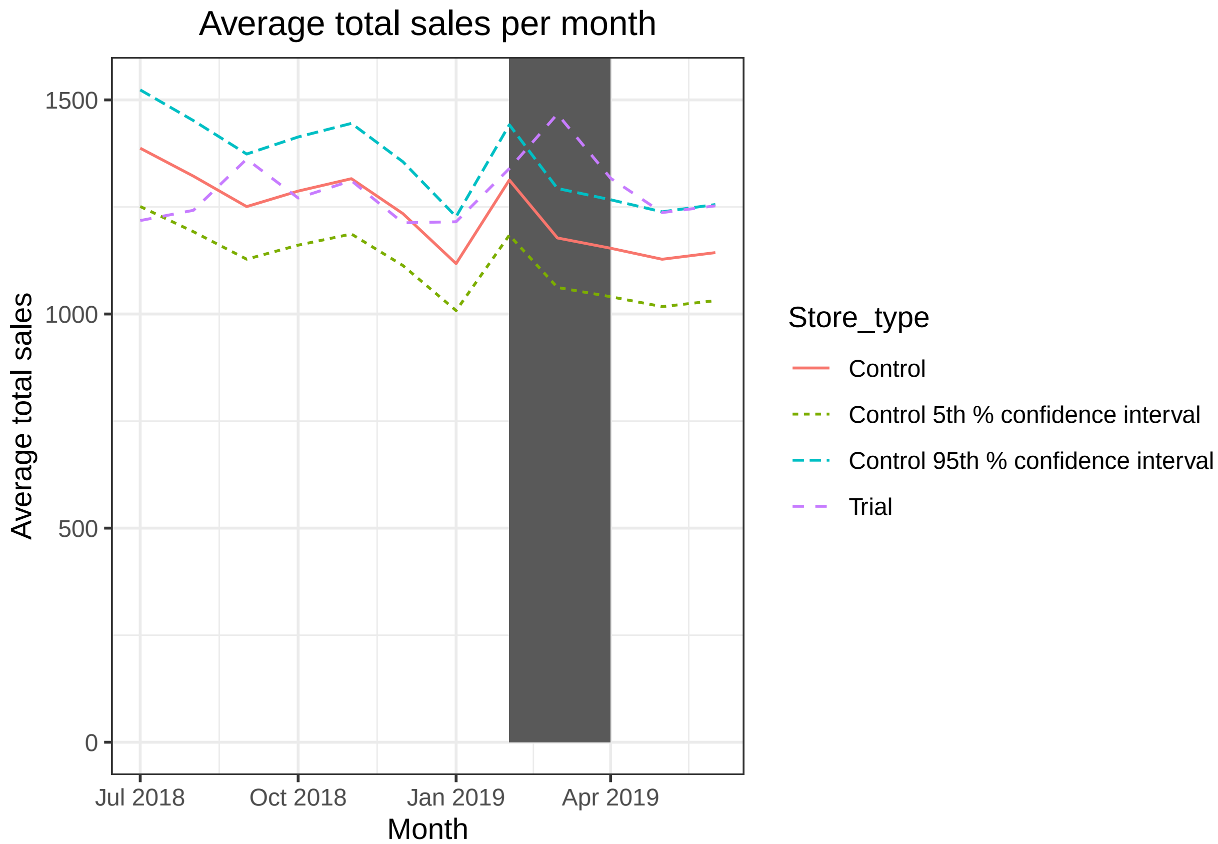
```
####  Conduct visual checks on trends based on the drivers
measureOverTimeCusts <- measureOverTime

pastCustomers88 <- measureOverTimeCusts[,
                  Store_type := ifelse(STORE_NBR == trial_store88, "Trial",
                  ifelse(STORE_NBR == control_store88, "Control", "Other stores"))][,
                  numberCustomers := mean(nCustomers), by = c("YEARMONTH", "Store_type")][,
                                                    YEARMONTH %% 100, 1, sep = "-"), "%Y-%m-%d")][YEARM

ggplot(pastCustomers88, aes(TransactionMonth, numberCustomers, color = Store_type)) +
```

```
  geom_line(aes(linetype = Store_type)) +
  labs(x = "Month", y = "Total customers",
       title = "Average total customers by month for stores 86/155 ")
```

Let's now assess the impact of the trial on sales.

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

scalingFactorForControlSales88 <- preTrialMeasures[STORE_NBR == trial_store88 & YEARMONTH < 201902,
sum(totSales)]/preTrialMeasures[STORE_NBR == control_store88 & YEARMONTH < 201902, sum(totSales)]

#### Apply the scaling factor
measureOverTimeSales <- measureOverTime

scaledControlSales88 <- measureOverTimeSales[STORE_NBR == control_store88,
                       ][, controlSales := totSales *scalingFactorForControlSales88]

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

#### Hint: When calculating percentage difference, remember to use absolute difference

percentageDiff88 <- merge(scaledControlSales88[, c("YEARMONTH", "controlSales")],
                   measureOverTime[STORE_NBR == trial_store88, c("YEARMONTH", "totSales")],

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

```r
#### Calculate the standard deviation of percentage differences during the pre-trial period

stdDev88 <- sd(percentageDiff88[YEARMONTH < 201902 , percentageDiff])

degreesOfFreedom <- 7

#### Trial and control store total sales
measureOverTimeSales <- measureOverTime

pastSales88 <- measureOverTimeSales[,
               Store_type := ifelse(STORE_NBR == trial_store88, "Trial",
               ifelse(STORE_NBR == control_store88, "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 %

#### Calculate the 5th and 95th percentile for control store sales.
#### Hint: The 5th and 95th percentiles can be approximated by using two standard deviations away from t

pastSales88_Controls95 <- pastSales88[Store_type == "Control",][,
                          totSales := totSales * (1 + stdDev88 * 2)][,
                          Store_type := "Control 95th % confidence interval"]

pastSales88_Controls5 <- pastSales88[Store_type == "Control",][,
                        totSales := totSales * (1 - stdDev88 * 2)][,
                        Store_type := "Control 5th % confidence interval"]

#### Hint2: Recall that the variable stdDev earlier calculates standard deviation in percentages, and no

#### Then, create a combined table with columns from pastSales, pastSales_Controls95 and pastSales_Contr

trialAssessmentSales88 <- rbind(pastSales88, pastSales88_Controls5, pastSales88_Controls95)
#### Plotting these in one nice graph

ggplot(trialAssessmentSales88, aes(TransactionMonth, totSales, color = Store_type)) +
  geom_rect(data = trialAssessmentSales88[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", y = "Average total sales", title = "Average total sales per 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.

```
#### This would be a repeat of the steps before for total sales
#### Scale pre-trial control customers to match pre-trial trial store customers

scalingFactorForControlCust88 <- preTrialMeasures[
  STORE_NBR == 88 & YEARMONTH < 201902, sum(nCustomers)]/preTrialMeasures[
  STORE_NBR == control_store86 & YEARMONTH < 201902, sum(nCustomers)]

#### Apply the scaling factor measureOverTimeCusts <- measureOverTime

scaledControlCustomers88 <- measureOverTimeCusts[STORE_NBR == control_store88,

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

percentageDiffcus88 <- merge(scaledControlCustomers88[, c("YEARMONTH","controlCustomers")],
  by = "YEARMONTH")[, percentageDiff := abs(controlCustomers-nCustomers)/controlCustomers]

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

stdDevcus88 <- sd(percentageDiffcus88[YEARMONTH < 201902 , percentageDiff])

degreesOfFreedom <- 7
```

```r
#### Trial and control store number of customers

pastCustomers88 <- measureOverTimeCusts[, nCusts := mean(nCustomers),
                  by = c("YEARMONTH", "Store_type")][Store_type %in% c("Trial", "Control"), ]

#### Control store 95th percentile

pastCustomers88_Controls95 <- pastCustomers88[Store_type == "Control",][,
                              nCusts := nCusts * (1 + stdDevcus88 * 2)][,
                              Store_type := "Control 95th % confidence interval"]

#### Control store 5th percentile
pastCustomers88_Controls5 <- pastCustomers88[Store_type == "Control",][,
                            nCusts := nCusts * (1 - stdDevcus88 * 2)][,
                            Store_type := "Control 5th % confidence interval"]

trialAssessmentcus88 <- rbind(pastCustomers88, pastCustomers88_Controls95, pastCustomers88_Controls5)

#### Plotting these in one nice graph

ggplot(trialAssessmentcus88, aes(TransactionMonth, nCusts, color = Store_type)) +  geom_rect(data = tri
aes(xmin = min(TransactionMonth), xmax = max(TransactionMonth),  ymin = 0 , ymax = Inf, color = NULL), s
```