Recast : Technical Exercise

Taiwo Owoseni

29/12/2022

Instructions

Clean and join the two datasets so that there is 1 column for each variable (channels and revenue) and 1 row for each date.

a. Please make note of any concerns you have with the data (ex. missing data, potential data errors etc.)

b. Export this to a CSV file

Exercise

1. Provide answers to the below questions with visualizations. Note: there may be numerous ways to interpret the questions below and therefore there are not necessarily "correct" answers to some of these questions.

a. Which channel had the most spend in 2022?

b. Which channel had the largest increase in spend so far in 2022 compared to the same date range in 2021?

c. Did Acme spend more with Google or Facebook in 2022? How has this changed since 2021?

d. Which retailer (DTC, Amazon or Walmart) accounted for the most revenue in October 2022?

e. In terms of total revenue, are there any anomalous days?

f. In which month of the year does Acme tend make the most revenue?

g. Does Acme's marketing spend tend to follow a similar pattern to revenue?

Import Libraries

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.4.0     v purrr   0.3.4
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1

## Warning: package 'ggplot2' was built under R version 4.1.2

## Warning: package 'tibble' was built under R version 4.1.2

## Warning: package 'tidyr' was built under R version 4.1.2

## Warning: package 'dplyr' was built under R version 4.1.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
library(ggplot2)
library(tidyverse)
library(scales)
```

```
## Warning: package 'scales' was built under R version 4.1.2
```

```
##
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:purrr':
##
##     discard
```

```
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

Reading the Data sets

```
acm.spend <- read_csv('acme_spend.csv') # read data
```

```
## New names:Rows: 10350 Columns: 4-- Column specification --------------------------------------------
## Delimiter: ","
## chr (2): date, channel
## dbl (2): ...1, spend
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
acm.rev <- read_csv('acme_revenue.csv') # read data
```

```
## New names:Rows: 1035 Columns: 5-- Column specification ---------------------------------------------
## Delimiter: ","
## chr (1): date
## dbl (4): ...1, revenue_dtc, revenue_amazon, revenue_walmart
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(acm.spend)
```

```
## # A tibble: 6 x 4
##    ...1 date   channel                  spend
##   <dbl> <chr>  <chr>                    <dbl>
## 1     1 1/1/20 facebook_prospecting     4154.
## 2     2 1/1/20 facebook_retargeting     2679.
## 3     3 1/1/20 google_branded_search     474 
## 4     4 1/1/20 google_nonbranded_search 3817.
## 5     5 1/1/20 pinterest                7421.
## 6     6 1/1/20 twitter                  2233.
```

```
head(acm.rev) # first 5 data points
```

```
## # A tibble: 6 x 5
##    ...1 date   revenue_dtc revenue_amazon revenue_walmart
##   <dbl> <chr>        <dbl>          <dbl>           <dbl>
## 1     1 1/1/20      16000           50000           34000 
## 2     2 1/2/20      19306.          60332           41026.
## 3     3 1/3/20      22052.          68914.          46861.
## 4     4 1/4/20      23177.          72428.          49251.
## 5     5 1/5/20      26169.          81778.          55609.
## 6     6 1/6/20      29430.          91969.          62539.
```

```
dim(acm.rev) # data dimension
```

```
## [1] 1035    5
```

```
dim(acm.spend) # data dimension
```

```
## [1] 10350     4
```

```
summary(acm.spend) # basic descriptive details
```

```
##       ...1           date              channel               spend      
##  Min.   :    1   Length:10350       Length:10350       Min.   :    0  
##  1st Qu.: 2588   Class :character   Class :character   1st Qu.: 1323  
##  Median : 5176   Mode  :character   Mode  :character   Median : 5358  
##  Mean   : 5176                                         Mean   : 7792  
##  3rd Qu.: 7763                                         3rd Qu.:11711  
##  Max.   :10350                                         Max.   :36401  
##                                                        NA's   :69     
```

Display Missing values in Revenue Data

```
null.rev<- acm.rev%>%filter_all(any_vars(is.na(.))) # find empty entries
null.rev
```

```
## # A tibble: 4 x 5
##    ...1 date     revenue_dtc revenue_amazon revenue_walmart
##   <dbl> <chr>          <dbl>          <dbl>           <dbl>
## 1    18 1/18/20           NA        163546.         111211.
## 2   213 7/31/20       35993.             NA          76486.
## 3  1012 10/8/22       58158.        181745.              NA 
## 4  1021 10/17/22          NA             NA         136361.
```

Drop missing Values

```
 acm.rev<- acm.rev |>
          select(-c(...1)) |> # drop column `...1`
          na.omit() # drop nan values

acm.spend<- acm.spend|>
           select(-c(...1))|> # drop column `...1`
           na.omit() # drop nan values

acm.rev
```

```
## # A tibble: 1,031 x 4
##    date    revenue_dtc revenue_amazon revenue_walmart
##    <chr>         <dbl>          <dbl>           <dbl>
##  1 1/1/20       16000         50000           34000
##  2 1/2/20       19306.        60332.          41026.
##  3 1/3/20       22052.        68914.          46861.
##  4 1/4/20       23177.        72428.          49251.
##  5 1/5/20       26169.        81778.          55609.
##  6 1/6/20       29430.        91969.          62539.
##  7 1/7/20       32550.       101720.          69170.
##  8 1/8/20       34248.       107026.          72778.
##  9 1/9/20       36125.       112892.          76766.
## 10 1/10/20      38360.       119875.          81515.
## # ... with 1,021 more rows
```

```
summary(acm.rev)
```

```
##      date            revenue_dtc     revenue_amazon   revenue_walmart
##  Length:1031        Min.   : 16000   Min.   : 50000   Min.   : 34000
##  Class :character   1st Qu.: 41717   1st Qu.:130364   1st Qu.: 88648
##  Mode  :character   Median : 72252   Median :225787   Median :153535
##                     Mean   : 72932   Mean   :227608   Mean   :154773
##                     3rd Qu.: 97568   3rd Qu.:304187   3rd Qu.:206847
##                     Max.   :193005   Max.   :415952   Max.   :282848
```

```
# convert channel to a factor
acm.spend$channel<-as.factor(acm.spend$channel) # change datatype to factor
acm.spend$date <- as.Date(acm.spend$date, "%m/%d/%y") # change datatype to date
acm.rev$date <-as.Date(acm.rev$date, "%m/%d/%y") # change datatype to date
```

```
# reshape data
acm.spend.wide <- pivot_wider(acm.spend, names_from = "channel", values_from ="spend")
```

```
#check for null values
acm.spend.wide%>% filter_all(any_vars(is.na(.)))
```

```
## # A tibble: 69 x 11
##    date       facebook_~1 faceb~2 googl~3 googl~4 pinte~5 twitter tiktok linea~6
##    <date>           <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl>   <dbl>
##  1 2022-08-24       9612.   4678.     460   6224.      NA   2514.   737.   7501.
##  2 2022-08-25       8870.   2740.     445   6550.      NA   2459.   940.   6531.
##  3 2022-08-26       9623.   3028.     507   6371.      NA   2833.  1353.   5691.
##  4 2022-08-27       7071.   4144.     443   5140.      NA   2283.  1314.   4686.
##  5 2022-08-28      11160.   2666.     485   4421.      NA   1729.  1067.   7671.
##  6 2022-08-29      10876.   4373.     598   5074.      NA   3154.   710.   5185.
##  7 2022-08-30       9756.   3334.     537   3572.      NA   2995.   800.   6440.
##  8 2022-08-31       9625.   3312.     414   4999.      NA   1628.  1390.   5201.
##  9 2022-09-01       9509.   4550.     577   3292.      NA   2744.   837.   3767.
## 10 2022-09-02       7431.   3918.     409   4993.      NA   1852.   954.   6615.
## # ... with 59 more rows, 2 more variables: online_display <dbl>,
## #   online_video <dbl>, and abbreviated variable names 1: facebook_prospecting,
## #   2: facebook_retargeting, 3: google_branded_search,
## #   4: google_nonbranded_search, 5: pinterest, 6: linear_tv
```

Joining the data sets

```
# left join by date
df<- left_join(acm.rev, acm.spend.wide, by="date")
df
```

```
## # A tibble: 1,031 x 14
##    date       revenue_~1 reven~2 reven~3 faceb~4 faceb~5 googl~6 googl~7 pinte~8
##    <date>          <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 2020-01-01     16000   50000   34000    4154.   2679.     474   3817.   7421.
##  2 2020-01-02     19306.  60332   41026.   9165.   3009.     505   2958.   5605.
##  3 2020-01-03     22052.  68914.  46861.   4383.   2511.     483   5388.  10587.
##  4 2020-01-04     23177.  72428.  49251.   9311.   3824.     528   3792.  10376.
##  5 2020-01-05     26169.  81778.  55609.   8487.   3873.     462   4053.   9356.
##  6 2020-01-06     29430.  91969.  62539.   7050.   4469.     436   5092.   7033.
##  7 2020-01-07     32550. 101720.  69170.   9949.   2376.     552   6218.  11112.
##  8 2020-01-08     34248. 107026.  72778.  10412.   3266.     469   5705.  13820.
##  9 2020-01-09     36125. 112892.  76766.  10313.   4227.     577   3639.  10056.
## 10 2020-01-10     38360. 119875.  81515.   8833.   4738.     569   4400.   9861.
## # ... with 1,021 more rows, 5 more variables: twitter <dbl>, tiktok <dbl>,
## #   linear_tv <dbl>, online_display <dbl>, online_video <dbl>, and abbreviated
## #   variable names 1: revenue_dtc, 2: revenue_amazon, 3: revenue_walmart,
## #   4: facebook_prospecting, 5: facebook_retargeting, 6: google_branded_search,
## #   7: google_nonbranded_search, 8: pinterest
```

```
summary(df)
```

```
##       date             revenue_dtc     revenue_amazon   revenue_walmart
##  Min.   :2020-01-01   Min.   : 16000   Min.   : 50000   Min.   : 34000
##  1st Qu.:2020-09-16   1st Qu.: 41717   1st Qu.:130364   1st Qu.: 88648
##  Median :2021-06-01   Median : 72252   Median :225787   Median :153535
##  Mean   :2021-05-31   Mean   : 72932   Mean   :227608   Mean   :154773
##  3rd Qu.:2022-02-13   3rd Qu.: 97568   3rd Qu.:304187   3rd Qu.:206847
##  Max.   :2022-10-31   Max.   :193005   Max.   :415952   Max.   :282848
##
##  facebook_prospecting facebook_retargeting google_branded_search
##  Min.   : 3215        Min.   : 1108        Min.   :400.0
##  1st Qu.:10941        1st Qu.: 4315        1st Qu.:451.0
##  Median :16268        Median : 6218        Median :498.0
##  Mean   :16976        Mean   : 6550        Mean   :500.0
##  3rd Qu.:21804        3rd Qu.: 8588        3rd Qu.:550.5
##  Max.   :33854        Max.   :13286        Max.   :600.0
##
##  google_nonbranded_search   pinterest         twitter            tiktok
##  Min.   : 1510            Min.   : 3094   Min.   : 822.1   Min.   : 366.9
##  1st Qu.: 5772            1st Qu.:11734   1st Qu.:2888.8   1st Qu.:1312.8
##  Median : 8447            Median :17713   Median :4007.4   Median :1897.5
##  Mean   : 8851            Mean   :18178   Mean   :4248.2   Mean   :1974.0
##  3rd Qu.:11433            3rd Qu.:23214   3rd Qu.:5444.5   3rd Qu.:2596.9
##  Max.   :17819            Max.   :35530   Max.   :8619.3   Max.   :3923.1
##                           NA's   :67
##    linear_tv     online_display     online_video
##  Min.   :    1   Min.   : 204.3   Min.   :    0
##  1st Qu.: 6834   1st Qu.: 683.0   1st Qu.:    0
##  Median : 9903   Median : 987.9   Median : 8982
##  Mean   :10309   Mean   :1040.8   Mean   :10071
##  3rd Qu.:13333   3rd Qu.:1357.5   3rd Qu.:17798
##  Max.   :20950   Max.   :2118.5   Max.   :36401
##
```

```r
null.data <- df%>% filter_all(any_vars(is.na(.)))
null.data
```

```
## # A tibble: 67 x 14
##    date       revenue_~1 reven~2 reven~3 faceb~4 faceb~5 googl~6 googl~7 pinte~8
##    <date>          <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 2022-08-24     35354. 110480.  75126.   9612.   4678.     460   6224.      NA
##  2 2022-08-25     35012. 109414.  74401.   8870.   2740.     445   6550.      NA
##  3 2022-08-26     35841. 112004.  76163.   9623.   3028.     507   6371.      NA
##  4 2022-08-27     37231. 116346.  79115.   7071.   4144.     443   5140.      NA
##  5 2022-08-28     36326. 113520.  77193.  11160.   2666.     485   4421.      NA
##  6 2022-08-29     36025. 112578.  76553.  10876.   4373.     598   5074.      NA
##  7 2022-08-30     36789. 114966.  78177.   9756.   3334.     537   3572.      NA
##  8 2022-08-31     38359. 119873.  81513.   9625.   3312.     414   4999.      NA
##  9 2022-09-01     39449. 123279.  83830.   9509.   4550.     577   3292.      NA
## 10 2022-09-02     39289. 122779.  83490.   7431.   3918.     409   4993.      NA
## # ... with 57 more rows, 5 more variables: twitter <dbl>, tiktok <dbl>,
## #   linear_tv <dbl>, online_display <dbl>, online_video <dbl>, and abbreviated
## #   variable names 1: revenue_dtc, 2: revenue_amazon, 3: revenue_walmart,
## #   4: facebook_prospecting, 5: facebook_retargeting, 6: google_branded_search,
## #   7: google_nonbranded_search, 8: pinterest
```

```r
# replace NAN with 0
df <- replace(df, is.na(df), 0)
null.data <- df%>% filter_all(any_vars(is.na(.)))
null.data
```

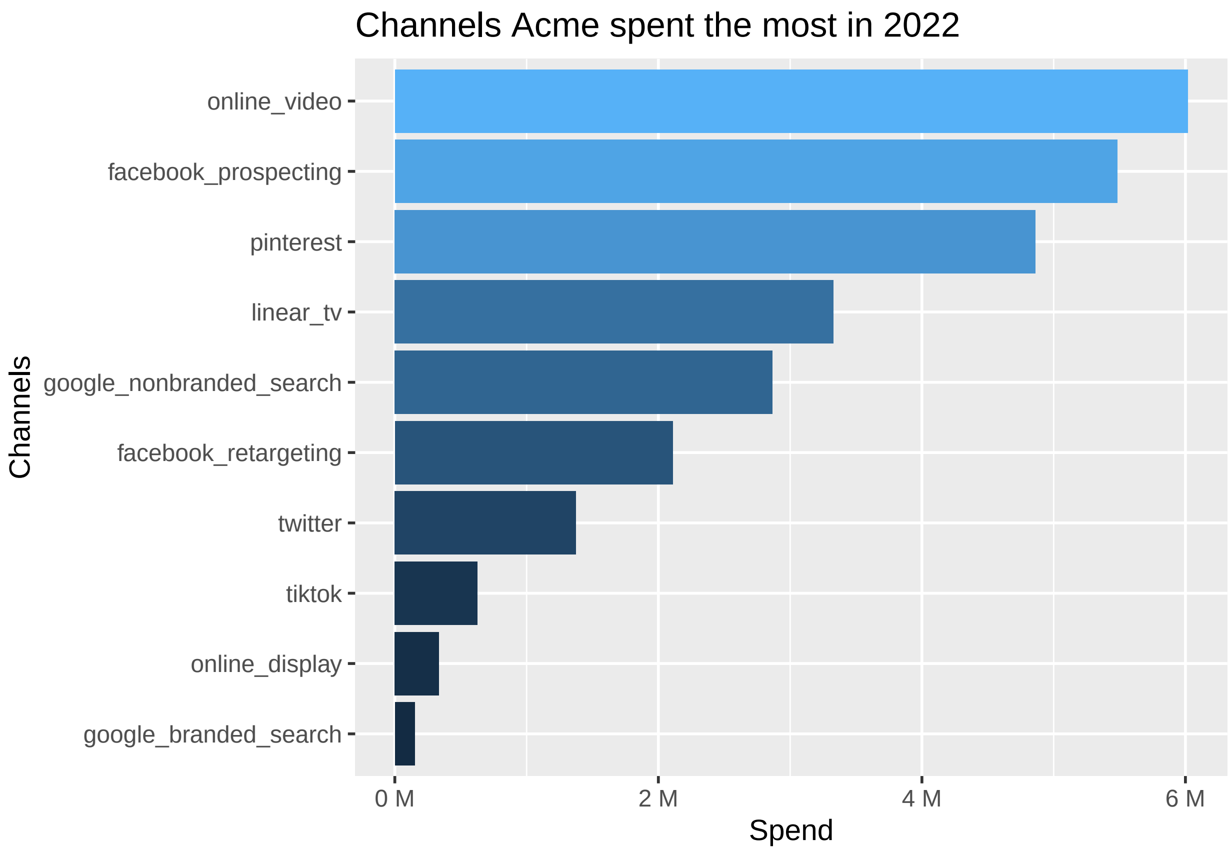
```
## # A tibble: 0 x 14
## # ... with 14 variables: date <date>, revenue_dtc <dbl>, revenue_amazon <dbl>,
## #   revenue_walmart <dbl>, facebook_prospecting <dbl>,
## #   facebook_retargeting <dbl>, google_branded_search <dbl>,
## #   google_nonbranded_search <dbl>, pinterest <dbl>, twitter <dbl>,
## #   tiktok <dbl>, linear_tv <dbl>, online_display <dbl>, online_video <dbl>
```

Which channel had the most spend in 2022

Observation : Acme spent about 6,000,000 on online videos in 2022

```r
df$year<- as.factor(format(df$date, format="%Y")) # create year column as factor
df$month_no <-  as.factor(format(df$date, format="%m")) # create month column as factor
df$month<- with(df, month.abb[month_no])
```

```
df|>
  filter(year == 2022)|> # filter on 2022
  summarize_at(vars(facebook_prospecting:online_video), sum)|> # summ up channels
  pivot_longer(cols = c("facebook_prospecting":"online_video"), # pvot the data longer
  names_to = "channels", values_to="spend")|>
  ggplot(aes( y= reorder(channels, spend), x=spend, fill=spend)) + # plot
  theme(legend.position="none") +
  scale_x_continuous(labels = unit_format(unit = "M", scale = 1e-6))+ # change the X scale
  geom_bar(stat = "identity") + # bar plot identity bar
  labs(title = "Channels Acme spent the most in 2022")+
  xlab("Spend") +
  ylab("Channels")
```



Which channel had the largest increase in spend so far in 2022 compared to the same date range in 2021?

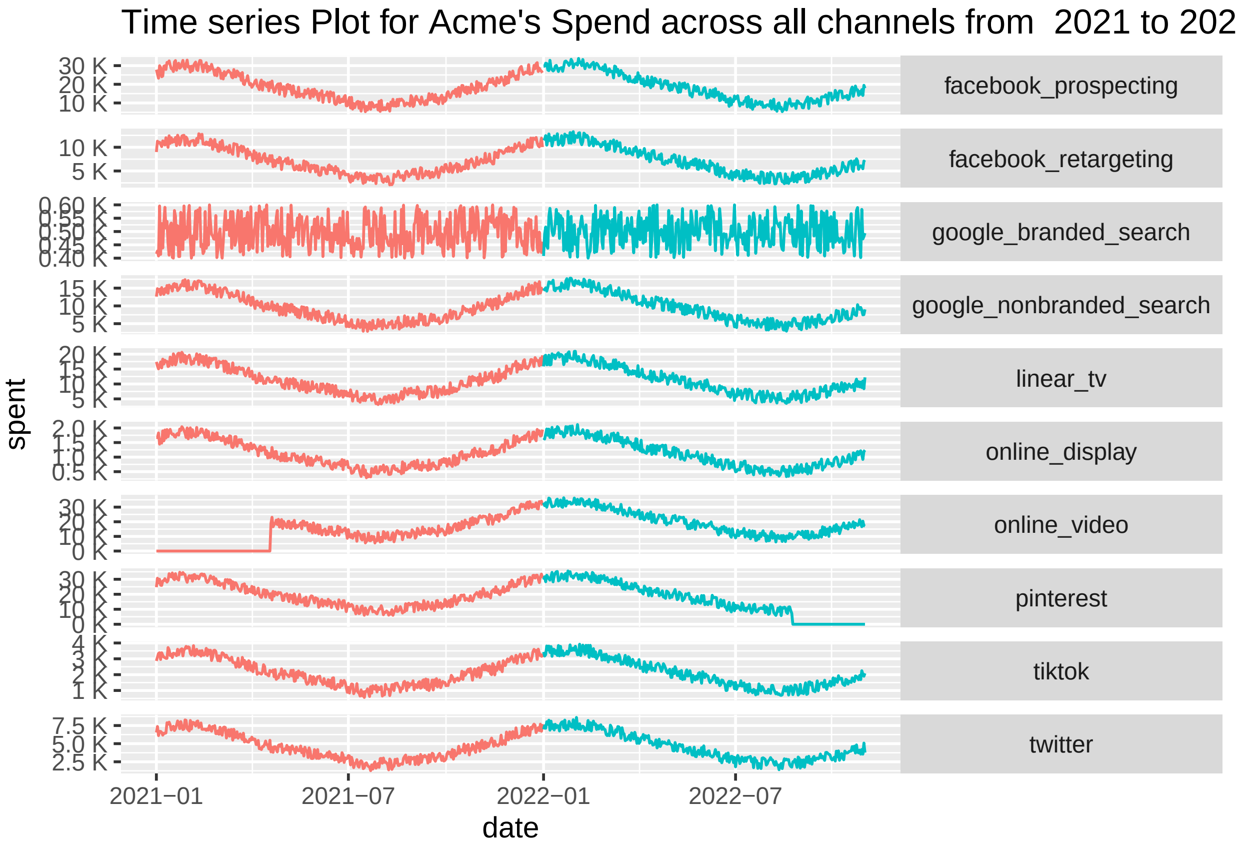
Observation: Online videos. From the plot, we can deduce that in the first quarter of 2021 there is no spending on online videos. However, in first quarter of 2022 it increased significantly. This is the highest increase in spending from 2021 to 2022.

```
channels<-names(df)[5:14]
colums <- c("date","year", channels)

monthly.channel <- df%>%
   filter(year %in% c(2021, 2022))|>
```

```
select(colums)
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect
## 1.1.0.

## Warning: Please use `all_of()` or `any_of()` instead.

## Warning: # Was:

## Warning: data %>% select(colums)

## Warning:

## Warning: # Now:

## Warning: data %>% select(all_of(colums))

## Warning:

## Warning: See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
```

```
monthly.channel<- pivot_longer(monthly.channel,
                          cols = names(monthly.channel)[3:12],
                           names_to = "channels", values_to = "spent")

monthly.channel|>
  ggplot(aes(y = spent, x = date))+
  geom_line(aes(color = year))+
  theme(legend.position = "none") +
  labs(title = "Time series Plot for Acme's Spend across all channels from  2021 to 2022 ")+
  scale_y_continuous(labels = unit_format(unit = "K", scale = 1e-3) )+
  theme(strip.text.y = element_text(angle = 0))+
  facet_grid(channels~., scale = "free")
```



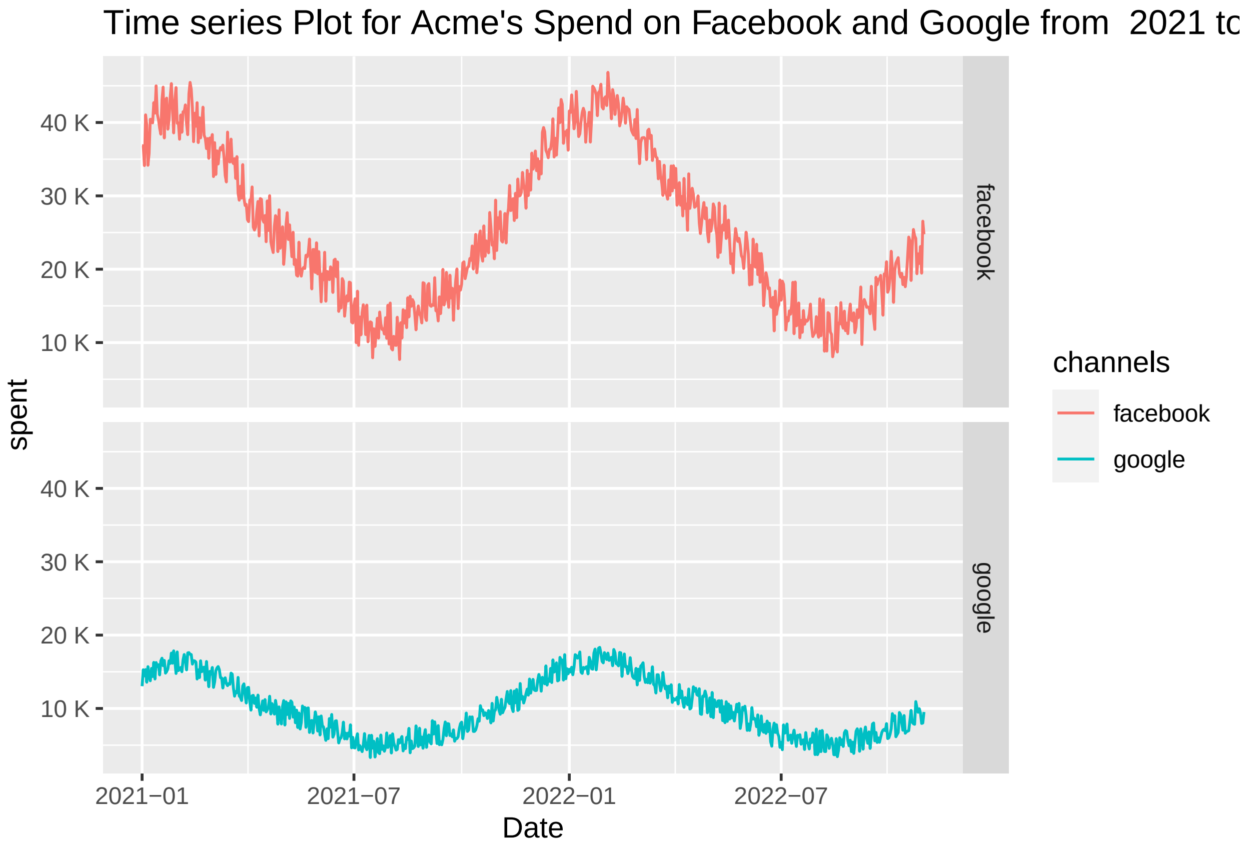
Did Acme spend more with Google or Facebook in 2022? How has this changed since 2021?

Observation: In 2022, Acme spent more with Facebook than Google. Additionally in 2021 and 2022, Acme spent over two times of google marketing on facebook marketing. Since the percentage spent on facebook is 72% and 28 % in 2022 and 2021 respectively, then there is no change

```
# source code for pattern matching : https://stackoverflow.com/questions/35693876/row-wise-sum-for-colu

channels<- c("facebook_prospecting", "facebook_retargeting", "google_branded_search","google_nonbranded_

colums <- c("date","year", channels)

ggle.fb.spend <- df%>%
   filter(year %in% c(2021, 2022))

ggle.fb.spend$google<- rowSums(ggle.fb.spend[grep('google', names(ggle.fb.spend)[-1])+1])

ggle.fb.spend$facebook<- rowSums(ggle.fb.spend[grep('facebook', names(ggle.fb.spend)[-1])+1])
```

```
ggle.fb.spend<-
  ggle.fb.spend|>
  select(c('google', 'facebook','date', 'year'))|>
  pivot_longer( cols = c('google', 'facebook'),
                names_to = "channels", values_to = "spent")
```

```
ggle.fb.spend|>
  ggplot(aes(y = spent, x = date, color = channels))+
  labs(title = "Time series Plot for Acme's Spend on Facebook and Google from  2021 to 2022 ")+
  geom_line()+
  scale_y_continuous(labels = unit_format(unit = "K", scale = 1e-3))+
  facet_grid(channels ~.)+
  xlab("Date")
```



```
summarise.gg.fb.spend<-ggle.fb.spend|>
  group_by(year, channels)|>
  summarise(total_spend = sum(spent))|>
  mutate(percent = total_spend /sum(total_spend))
```

```
## `summarise()` has grouped output by 'year'. You can override using the `.groups`
## argument.
```

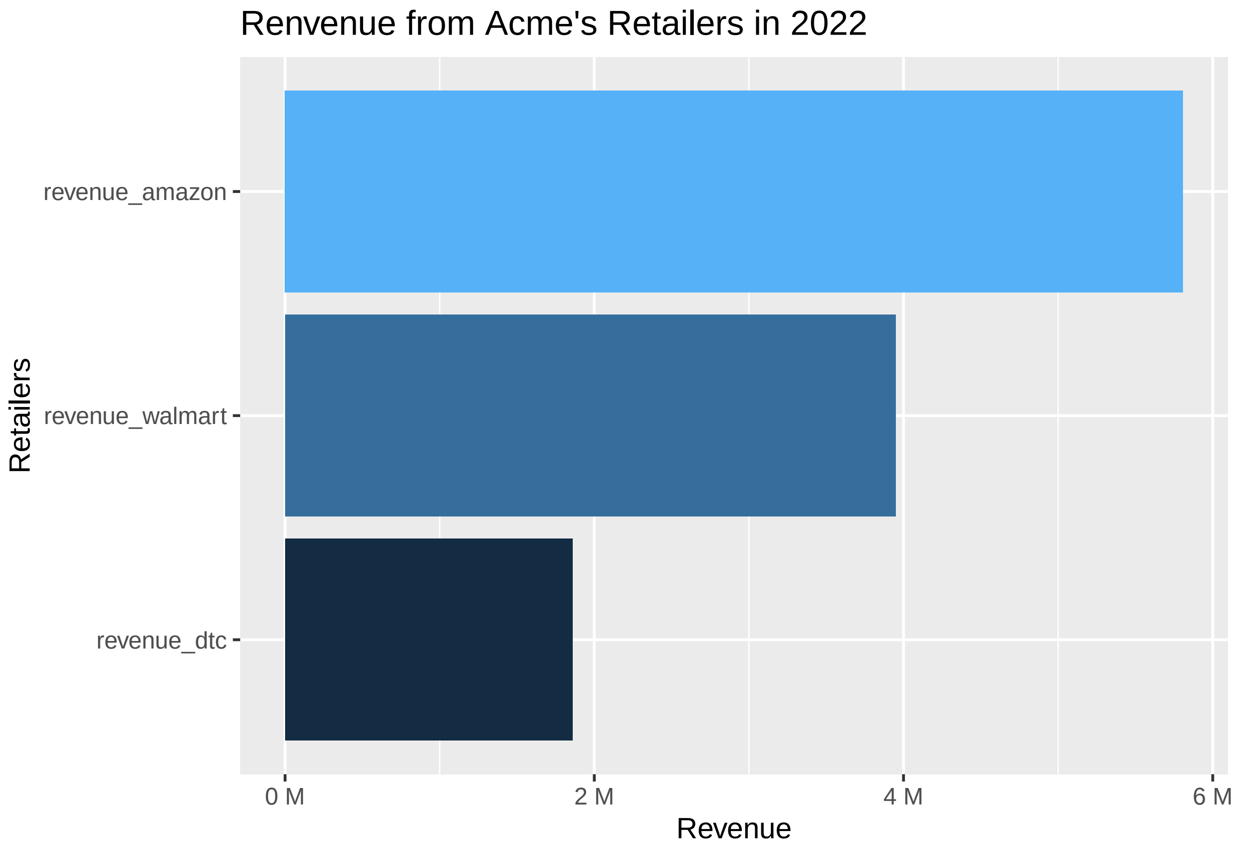
```
summarise.gg.fb.spend
```

```
## # A tibble: 4 x 4
## # Groups:   year [2]
##   year  channels total_spend percent
##   <fct> <chr>          <dbl>   <dbl>
## 1 2021  facebook    9440333.   0.718
## 2 2021  google      3712664.   0.282
## 3 2022  facebook    7591534.   0.716
## 4 2022  google      3015926.   0.284
```

Which retailer (DTC, Amazon or Walmart) accounted for the most revenue in October 2022?

Observation : Acme generated about 6,000,000 in revenue on Amazon

```
df|>
  filter(month=="Oct" & year == "2022")|> # filter on month and year
  summarize_at(vars(revenue_dtc: revenue_walmart), sum)|> # sum up across the revenue
  pivot_longer(cols = c("revenue_dtc": "revenue_walmart"), # puvot table
  names_to = "retailers", values_to="revenue")|>
  ggplot(aes( y= reorder(retailers, revenue), x=revenue, fill=revenue)) +
  theme(legend.position="none") +
  scale_x_continuous(labels = unit_format(unit = "M", scale = 1e-6))+ # change the X scale
  geom_bar(stat = "identity") + # bar plot idenitity bar
  labs(title = "Renvenue from Acme's Retailers in 2022")+
  xlab("Revenue") +
  ylab("Retailers")
```

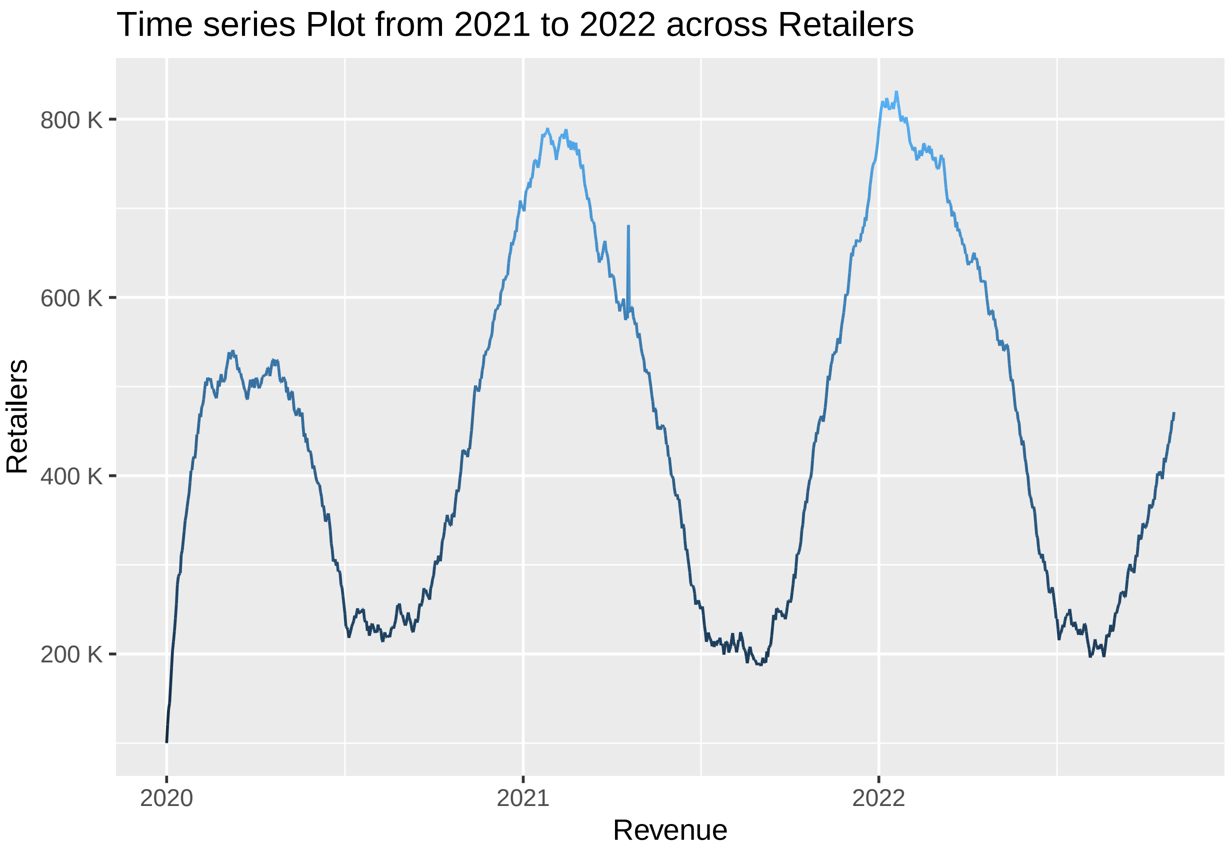


In terms of total revenue, are there any anomalous days?

Observation: In terms of total revenue, graphically, In April 19th, 2021 anomaly is present in DTC's revenue. I will filter it out

```
total.rev<-
  df|>
  select(date, revenue_walmart, revenue_amazon, revenue_dtc)|>
  mutate(total_rev = revenue_walmart +revenue_amazon +revenue_dtc)
```

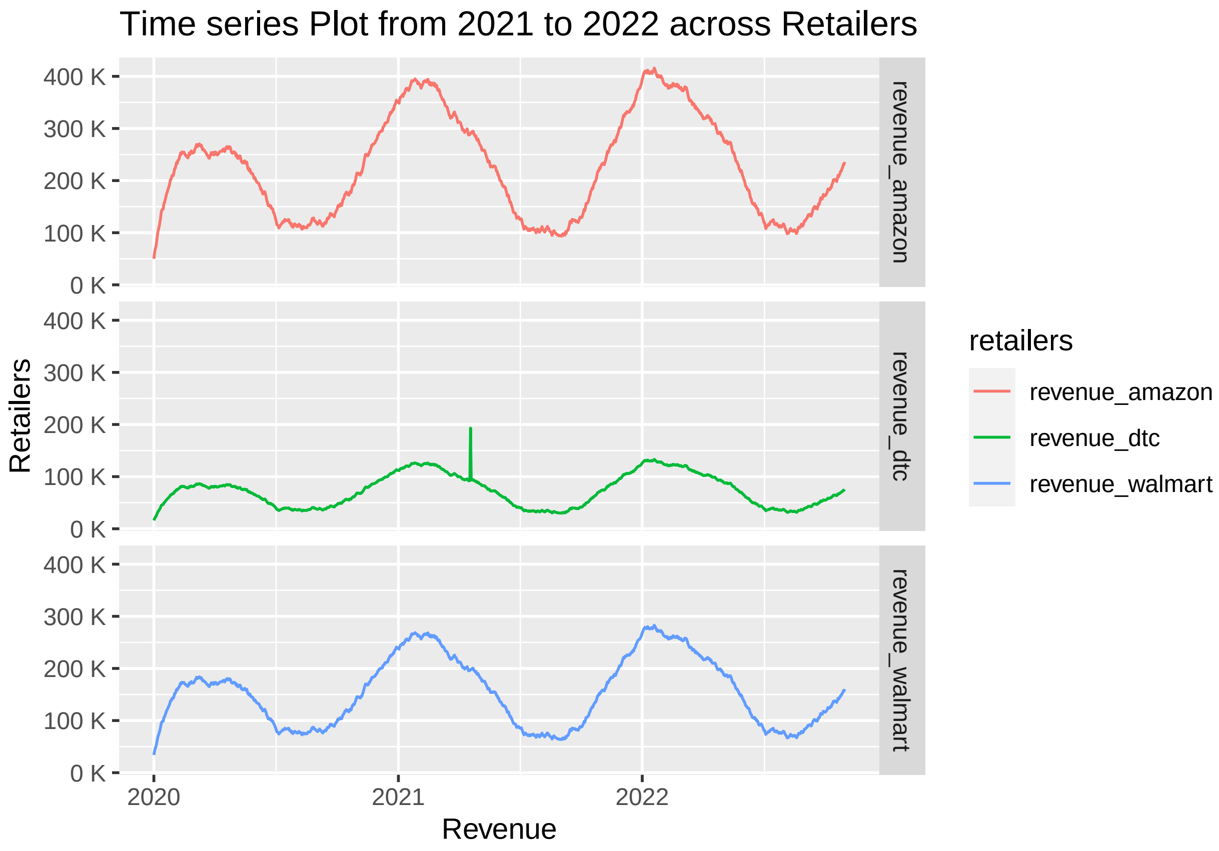
```
ts.plot <- total.rev|>
          ggplot(aes(x=date, y= total_rev, color = total_rev)) +
          geom_line() +
          labs(title = "Time series Plot from 2021 to 2022 across Retailers")+
          xlab("Revenue") +
          ylab("Retailers") +
          theme(legend.position = "none") +
          scale_y_continuous(labels = unit_format(unit = "K", scale = 1e-3))
ts.plot
```



```
revnue_list<-c('revenue_amazon', 'revenue_dtc', 'revenue_walmart','date','month')
longer.rev<-
  df|>select(revnue_list)|>
  pivot_longer( names_to = "retailers",
                values_to = "revenue",
                cols = c('revenue_amazon', 'revenue_dtc', 'revenue_walmart'))
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect
## 1.1.0.

## Warning: Please use `all_of()` or `any_of()` instead.

## Warning: # Was:

## Warning: data %>% select(revnue_list)
## Warning:

## Warning: # Now:

## Warning: data %>% select(all_of(revnue_list))

## Warning:

## Warning: See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
```

```
ts.plot <- longer.rev|>
          ggplot(aes(x=date, y= revenue, color= retailers)) +
          geom_line() +
          labs(title = "Time series Plot from 2021 to 2022 across Retailers")+
          xlab("Revenue") +
          ylab("Retailers") +
          scale_y_continuous(labels = unit_format(unit = "K", scale = 1e-3))+
          facet_grid( retailers ~ .)
ts.plot
```



```
rev.dtc.outliers <- boxplot(df$revenue_dtc, plot=FALSE)$out
df[df$revenue_dtc == rev.dtc.outliers, ]
```

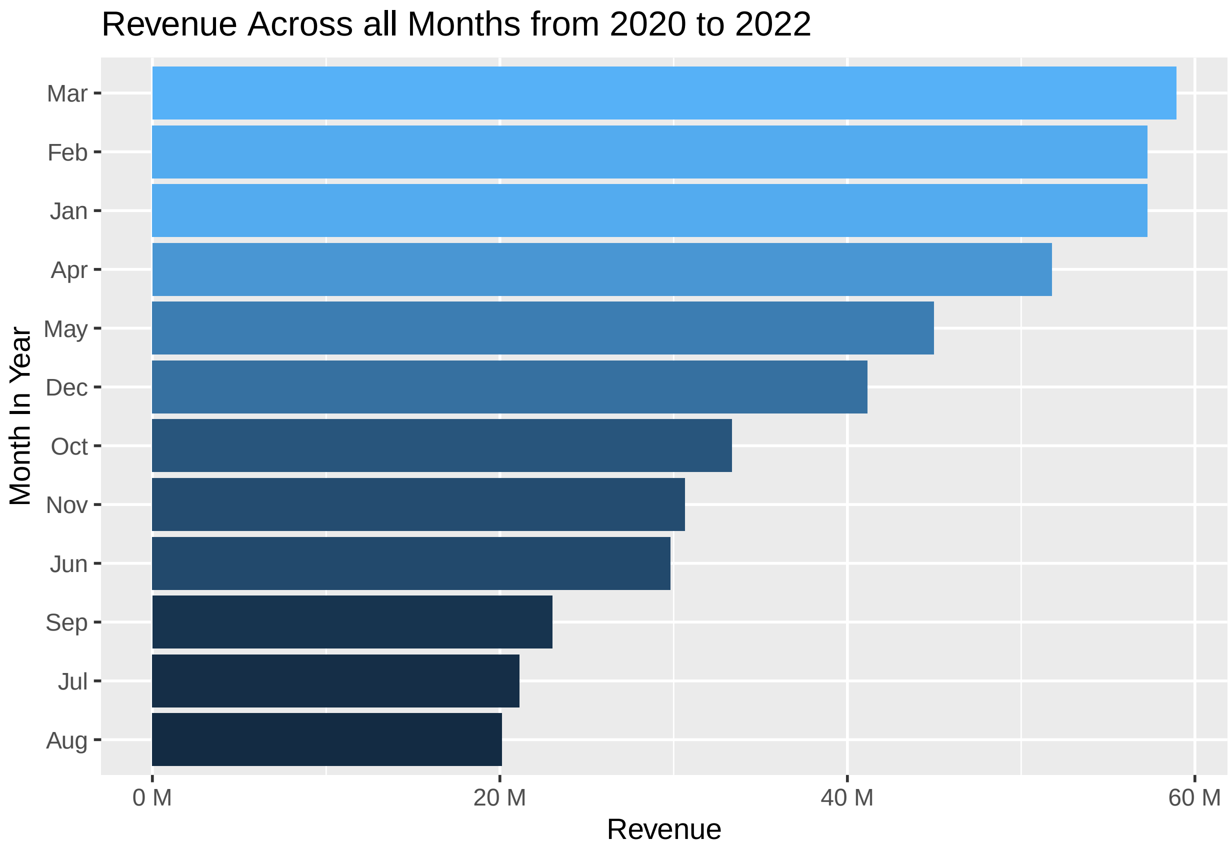
```
## # A tibble: 1 x 17
##   date       revenue_dtc reven~1 reven~2 faceb~3 faceb~4 googl~5 googl~6 pinte~7
##   <date>           <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 2021-04-19     193005. 290642. 197637.  18258.   7021.     451   9783.  20465.
## # ... with 8 more variables: twitter <dbl>, tiktok <dbl>, linear_tv <dbl>,
## #   online_display <dbl>, online_video <dbl>, year <fct>, month_no <fct>,
## #   month <chr>, and abbreviated variable names 1: revenue_amazon,
## #   2: revenue_walmart, 3: facebook_prospecting, 4: facebook_retargeting,
## #   5: google_branded_search, 6: google_nonbranded_search, 7: pinterest
```

```
df<- df[-which(df$revenue_dtc %in% rev.dtc.outliers),]
```

In which month of the year does Acme tend make the most revenue?

Observation: The total revenue for Acme In the month of March is the highest

```
monthly.rev <-longer.rev|>
  group_by(month)|>
  summarise(monthly_sum_revenue = sum(revenue))|>
  ggplot(aes( y= reorder(month, monthly_sum_revenue), x=monthly_sum_revenue, fill   =monthly_sum_revenu
  theme(legend.position="none") +
  scale_x_continuous(labels = unit_format(unit = "M", scale = 1e-6))+ # change the X scale
  geom_bar(stat = "identity") + # bar plot idenitity bar
  labs(title = "Revenue Across all Months from 2020 to 2022")+
  xlab("Revenue") +
  ylab("Month In Year") +
  geom_bar(stat = 'identity')

monthly.rev
```



Does Acme’s marketing spend tend to follow a similar pattern to revenue?–

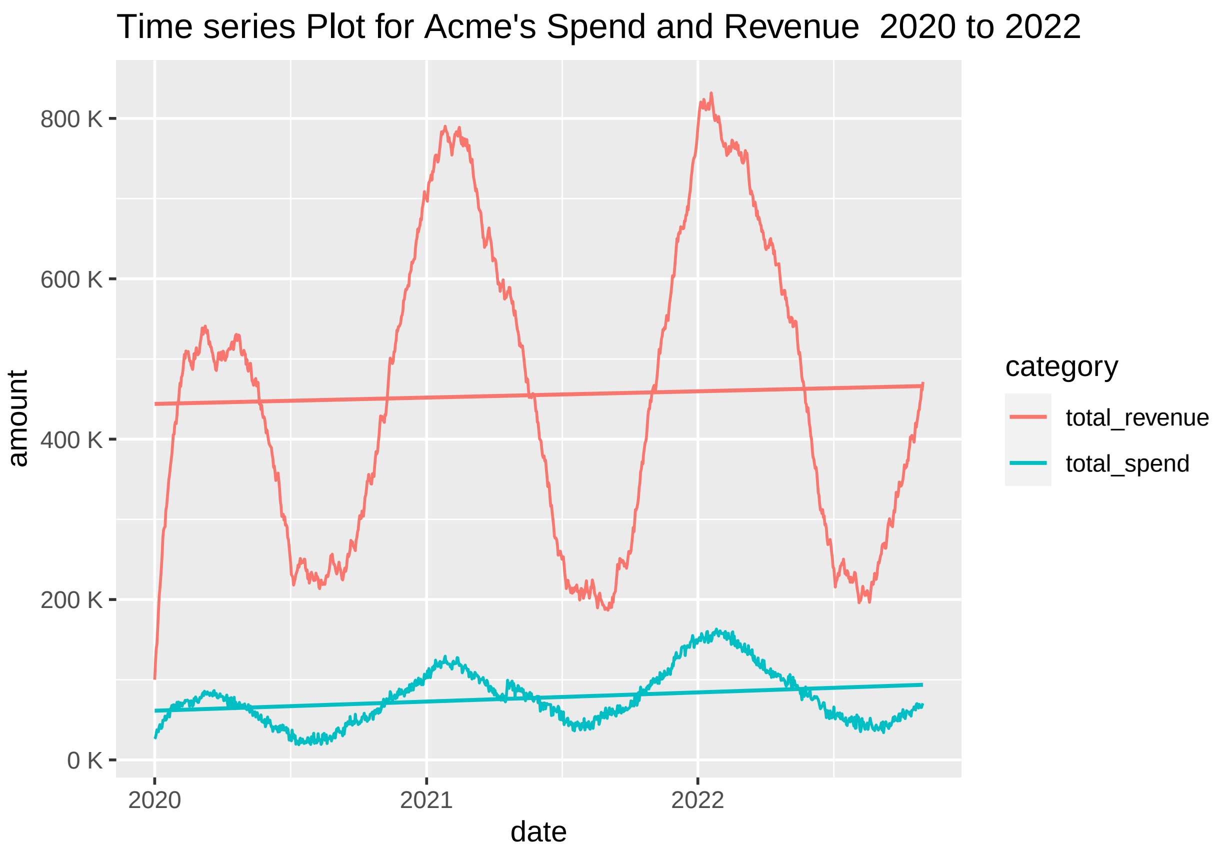
Graphically, the Acme’s Marketing Spend and Retail revenue tend to follow similar pattern. However, this is not enough to confidently say they are correlated or there is a causal relationship between the two.

```r
# total spend
# total revenuve

df$total_revenue <- rowSums(df[c(2:4)])
df$total_spend <- rowSums(df[c(5:14)])

total.spend.rev<- df[c(1,18,19)]
total.spend.rev
```

```
## # A tibble: 1,030 x 3
##    date       total_revenue total_spend
##    <date>             <dbl>       <dbl>
##  1 2020-01-01       100000       26297.
##  2 2020-01-02       120664       27722.
##  3 2020-01-03       137828.      32155.
##  4 2020-01-04       144856.      37625.
##  5 2020-01-05       163555.      35192.
##  6 2020-01-06       183939.      35031.
##  7 2020-01-07       203440.      41153.
##  8 2020-01-08       214052.      44104.
##  9 2020-01-09       225783.      39343.
## 10 2020-01-10       239750.      39211.
## # ... with 1,020 more rows
```

```r
total.spend.rev.long<-total.spend.rev|>
  pivot_longer(names_to = "category", cols = c('total_revenue', 'total_spend'), values_to= 'amount')

total.spend.rev.long|>
 ggplot(aes(y = amount, x = date, color=  category ))+
  geom_line()+
   labs(title = "Time series Plot for Acme's Spend and Revenue  2020 to 2022 ")+
  geom_smooth(method =lm, size = 0.7, se=FALSE)+
  scale_y_continuous(labels = unit_format(unit = "K", scale = 1e-3))
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.

## Warning: Please use `linewidth` instead.

## `geom_smooth()` using formula = 'y ~ x'
```


Time series Plot for Acme's Spend and Revenue 2020 to 2022
amount
800 K
600 K
400 K
200 K
0 K
2020
2021
2022
date
category
total_revenue
total_spend