Data Ingestion/EDA

Emilio Sanchez San Martin

2025-04-06

Continuing Data Investigation

First, I have to get every library that I will use for this Data Investigation/EDA.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
library(ggplot2)
```

I have to put the three datasets now on my global envrionment to work with them.

```
# Loading the datasets
High_Volume_Weekly_DS <- read.csv("/Users/emilio/Downloads/DATA 205/Montgomery-College-Data-Set-1(High-
Medium_Volume_Weekly_DS <- read.csv("/Users/emilio/Downloads/DATA 205/Montgomery-College-Data-Set-1(Medi
Low_Volume_Weekly_DS <- read.csv("/Users/emilio/Downloads/DATA 205/Montgomery-College-Data-Set-1(Low-Vo
```

I noticed that the variables on the top of my data set have X1, X2, X3, etc. I will remove the X and replace it with “Week”.

```
# Renaming the columns
colnames(High_Volume_Weekly_DS) <- gsub("X", "Week", colnames(High_Volume_Weekly_DS))
colnames(Medium_Volume_Weekly_DS) <- gsub("X", "Week", colnames(Medium_Volume_Weekly_DS))
colnames(Low_Volume_Weekly_DS) <- gsub("X", "Week", colnames(Low_Volume_Weekly_DS))
```

Finally, I will do my last step with this data set, which is to multiply to “Bottles_Per_Case” varibale with “Cost_Amount_Per_Bottle” variable to get the “Total_Cost” variable for all the bottles in a case. I will do this for all three data sets.

```r
# Creating the Total_Cost variable
High_Volume_Weekly_DS <- High_Volume_Weekly_DS |>
  mutate(Total_Cost = Bottles_Per_Case * Cost_Amount_Per_Bottle)

Medium_Volume_Weekly_DS <- Medium_Volume_Weekly_DS |>
  mutate(Total_Cost = Bottles_Per_Case * Cost_Amount_Per_Bottle)

Low_Volume_Weekly_DS <- Low_Volume_Weekly_DS |>
  mutate(Total_Cost = Bottles_Per_Case * Cost_Amount_Per_Bottle)
```

I want to move the last column "Total_Cost" right after the "Cost_Amount_Per_Bottle" column. I will do this for all three data sets.

```r
# Moving the Total_Cost column
High_Volume_Weekly_DS <- High_Volume_Weekly_DS |>
  select(ItemID, Description, Bottles_Per_Case, Cost_Amount_Per_Bottle, Total_Cost, everything())
Medium_Volume_Weekly_DS <- Medium_Volume_Weekly_DS |>
  select(ItemID, Description, Bottles_Per_Case, Cost_Amount_Per_Bottle, Total_Cost, everything())
Low_Volume_Weekly_DS <- Low_Volume_Weekly_DS |>
  select(ItemID, Description, Bottles_Per_Case, Cost_Amount_Per_Bottle, Total_Cost, everything())
```

Finished! I will do my last step, which is to check the structure of the data sets to see if everything is in order. I will do this by using the str() function.

```r
# Checking the structure of the datasets
str(High_Volume_Weekly_DS)
```

```
## 'data.frame':    500 obs. of  59 variables:
##  $ ItemID                : int  18467 96741 70417 249230 35211 53929 71009 359430 98744 71983 ...
##  $ Description           : chr  "SCOTTY'S VODKA 50ML" "CORONA EXTRA 4/6 NR - 12OZ" "FIREBALL CINN WHI
##  $ Bottles_Per_Case      : int  96 24 120 120 120 6 24 120 24 24 ...
##  $ Cost_Amount_Per_Bottle: num  0.99 1.83 1.09 1.09 0.99 ...
##  $ Total_Cost            : num  95 44 131 131 119 ...
##  $ Week1                 : int  60 162 62 230 142 92 48 96 120 40 ...
##  $ Week2                 : int  93 0 170 100 181 127 60 72 84 56 ...
##  $ Week3                 : int  93 0 170 100 181 127 60 72 84 56 ...
##  $ Week4                 : int  94 144 108 280 154 90 84 102 108 56 ...
##  $ Week5                 : int  89 144 166 210 167 153 96 132 0 88 ...
##  $ Week6                 : int  114 132 141 240 224 151 108 120 102 80 ...
##  $ Week7                 : int  145 252 203 170 152 129 24 54 132 56 ...
##  $ Week8                 : int  94 66 185 80 189 131 24 78 42 48 ...
##  $ Week9                 : int  63 6 153 20 260 134 48 138 36 72 ...
##  $ Week10                : int  130 0 145 70 215 127 120 78 0 72 ...
##  $ Week11                : int  68 66 133 120 191 111 132 114 84 88 ...
##  $ Week12                : int  153 126 160 200 200 129 72 78 84 48 ...
##  $ Week13                : int  90 168 143 200 197 131 48 174 138 56 ...
##  $ Week14                : int  85 84 141 150 147 135 132 120 72 32 ...
##  $ Week15                : int  71 156 192 150 205 134 84 120 90 152 ...
##  $ Week16                : int  132 162 158 110 231 99 60 84 138 120 ...
##  $ Week17                : int  127 108 241 0 151 132 0 156 102 72 ...
##  $ Week18                : int  172 138 157 0 139 117 0 114 78 72 ...
##  $ Week19                : int  94 150 219 0 154 106 0 168 120 200 ...
##  $ Week20                 : int  137 144 245 0 5 114 0 90 126 88 ...
##  $ Week21                 : int  117 120 246 90 0 152 132 72 18 200 ...
##  $ Week22                 : int  121 210 200 210 0 119 120 102 168 160 ...
##  $ Week23                 : int  96 150 167 150 92 140 252 114 120 192 ...
##  $ Week24                 : int  142 264 192 160 125 149 108 126 186 184 ...
##  $ Week25                 : int  114 24 127 220 84 120 240 78 30 224 ...
##  $ Week26                 : int  167 132 155 170 107 138 120 96 108 224 ...
##  $ Week27                 : int  159 180 140 200 51 144 252 132 198 184 ...
##  $ Week28                 : int  125 186 121 130 252 122 120 162 138 136 ...
##  $ Week29                 : int  142 186 95 180 123 142 180 108 126 80 ...
##  $ Week30                 : int  203 90 144 170 141 106 300 66 60 112 ...
##  $ Week31                 : int  165 144 141 120 172 119 240 78 96 72 ...
##  $ Week32                 : int  174 186 123 170 140 127 216 78 126 64 ...
##  $ Week33                 : int  249 84 129 140 24 105 120 72 108 120 ...
##  $ Week34                 : int  183 90 210 50 8 115 204 78 78 112 ...
##  $ Week35                 : int  205 168 97 160 48 132 276 84 102 176 ...
##  $ Week36                 : int  255 162 124 130 97 146 192 162 162 112 ...
##  $ Week37                 : int  225 174 127 240 142 138 120 90 96 96 ...
##  $ Week38                 : int  235 222 103 150 102 120 264 108 162 144 ...
##  $ Week39                 : int  233 150 142 140 131 109 84 60 126 72 ...
##  $ Week40                 : int  240 138 184 130 126 132 120 96 156 40 ...
##  $ Week41                 : int  233 144 221 160 97 126 36 96 78 56 ...
##  $ Week42                 : int  183 126 148 160 143 111 132 180 120 64 ...
##  $ Week43                 : int  168 222 20 140 163 134 324 186 114 104 ...
##  $ Week44                 : int  167 120 131 80 143 152 204 162 108 56 ...
##  $ Week45                 : int  164 180 152 210 160 144 204 114 150 128 ...
##  $ Week46                 : int  170 174 0 140 174 153 144 204 198 136 ...
##  $ Week47                 : int  159 216 0 120 113 140 84 216 54 48 ...
##  $ Week48                 : int  137 294 147 280 155 187 132 150 252 168 ...
##  $ Week49                 : int  99 102 81 150 165 128 60 162 84 184 ...
##  $ Week50                 : int  118 114 0 290 219 150 180 288 114 160 ...
##  $ Week51                 : int  163 126 71 0 157 143 168 420 144 104 ...
##  $ Week52                 : int  102 252 169 0 99 158 144 60 264 184 ...
##  $ Week53                 : int  76 114 0 0 45 94 156 0 66 168 ...
##  $ Grand_Total            : int  7690 7478 7520 7391 7404 6901 6855 6481 5876 5842 ...
```

```r
str(Medium_Volume_Weekly_DS)
```

```
## 'data.frame':    500 obs. of  59 variables:
##  $ ItemID                 : int  70417 96741 96067 96083 98744 69086 249230 5932 53929 70058 ...
##  $ Description            : chr  "FIREBALL CINN WHISKY 50ML/10PK LOOSE" "CORONA EXTRA 4/6 NR - 12OZ"
##  $ Bottles_Per_Case       : int  120 24 24 24 24 24 120 24 6 120 ...
##  $ Cost_Amount_Per_Bottle: num  1.09 1.83 1.83 1.83 2 ...
##  $ Total_Cost             : num  131 44 44 44 48 ...
##  $ Week1                  : int  76 60 12 96 18 48 20 24 24 17 ...
##  $ Week2                  : int  60 60 132 52 48 60 80 60 42 35 ...
##  $ Week3                  : int  137 78 24 0 0 12 10 6 47 31 ...
##  $ Week4                  : int  136 24 72 54 49 24 110 48 46 31 ...
##  $ Week5                  : int  115 48 96 85 0 48 100 12 35 36 ...
##  $ Week6                  : int  127 90 72 79 90 60 80 24 54 36 ...
##  $ Week7                  : int  81 78 60 102 72 36 130 60 56 39 ...
##  $ Week8                  : int  113 0 72 36 6 0 40 0 40 27 ...
##  $ Week9                  : int  118 48 54 18 24 24 30 24 42 18 ...
##  $ Week10                 : int  158 24 54 30 36 24 20 42 40 30 ...
##  $ Week11                : int  101 36 138 41 36 24 60 30 46 35 ...
##  $ Week12                : int  128 72 66 26 30 84 100 72 36 14 ...
##  $ Week13                : int  160 84 180 79 90 0 50 66 38 18 ...
##  $ Week14                : int  102 48 60 36 36 0 80 12 30 29 ...
##  $ Week15                : int  86 54 72 43 37 84 0 30 44 47 ...
##  $ Week16                : int  117 114 48 61 24 24 70 54 35 27 ...
##  $ Week17                : int  91 48 168 62 42 48 50 42 46 35 ...
##  $ Week18                : int  121 96 132 90 103 84 70 60 34 40 ...
##  $ Week19                : int  109 108 54 134 72 36 60 72 32 50 ...
##  $ Week20                : int  102 54 54 87 50 108 60 54 47 41 ...
##  $ Week21                : int  110 168 138 102 72 156 60 90 55 39 ...
##  $ Week22                : int  100 192 90 127 72 84 90 66 26 56 ...
##  $ Week23                : int  110 138 150 72 54 96 90 54 39 15 ...
##  $ Week24                : int  84 180 48 240 137 96 50 48 57 21 ...
##  $ Week25                : int  55 90 210 66 78 72 60 42 34 25 ...
##  $ Week26                : int  72 66 138 90 24 120 50 60 42 34 ...
##  $ Week27                : int  73 150 150 162 144 132 90 78 56 18 ...
##  $ Week28                : int  76 90 72 78 63 108 120 48 33 1 ...
##  $ Week29                : int  80 84 48 96 66 36 40 84 36 48 ...
##  $ Week30                : int  57 114 72 54 72 72 80 24 44 26 ...
##  $ Week31                : int  79 84 90 72 45 72 60 36 47 55 ...
##  $ Week32                : int  61 120 162 108 36 36 20 66 45 37 ...
##  $ Week33                : int  82 120 66 84 96 60 60 48 38 24 ...
##  $ Week34                : int  131 126 84 66 99 60 20 48 31 50 ...
##  $ Week35                : int  87 84 84 138 80 84 50 18 34 35 ...
##  $ Week36                : int  103 60 24 84 73 96 70 90 35 27 ...
##  $ Week37                : int  97 102 36 78 114 24 40 36 27 64 ...
##  $ Week38                : int  113 102 24 78 54 12 30 18 36 34 ...
##  $ Week39                : int  97 42 42 78 78 48 50 36 37 76 ...
##  $ Week40                : int  138 72 18 96 62 48 50 12 36 65 ...
##  $ Week41                : int  126 108 72 72 96 48 80 60 41 73 ...
##  $ Week42                : int  90 42 78 72 61 48 30 18 48 48 ...
##  $ Week43                : int  112 90 18 96 63 96 20 30 47 81 ...
##  $ Week44                : int  154 36 12 60 36 24 20 30 46 99 ...
##  $ Week45                : int  119 54 18 84 36 48 40 36 52 66 ...
##  $ Week46                : int  135 54 84 84 144 84 40 42 41 72 ...
##  $ Week47                : int  109 144 48 60 66 48 40 54 47 33 ...
##  $ Week48                : int  105 144 78 126 108 108 40 48 71 43 ...
##  $ Week49                : int  155 30 36 12 72 36 0 6 39 63 ...
##  $ Week50                : int  131 54 66 36 42 96 0 12 36 77 ...
##  $ Week51                : int  137 54 120 78 168 108 0 42 52 66 ...
##  $ Week52                : int  124 222 108 60 156 108 0 0 45 36 ...
##  $ Week53                : int  26 24 42 36 42 12 0 48 24 9 ...
##  $ Grand_Total           : int  5687 4490 4172 4082 3498 3230 2831 2246 2228 2273 ...
```

```r
str(Low_Volume_Weekly_DS)
```

```
## 'data.frame':    500 obs. of  59 variables:
##  $ ItemID                : int  70417 96741 96083 98744 251042 96067 35211 38067 12944 23193 ...
##  $ Description           : chr  "FIREBALL CINN WHISKY 50ML/10PK LOOSE" "CORONA EXTRA 4/6 NR - 12OZ"
##  $ Bottles_Per_Case      : int  120 24 24 24 120 24 120 120 24 24 ...
##  $ Cost_Amount_Per_Bottle: num  1.09 1.83 1.83 2 1.09 ...
##  $ Total_Cost            : num  131 44 44 48 131 ...
##  $ Week1                 : int  141 66 48 60 0 60 64 18 36 12 ...
##  $ Week2                : int  88 78 90 78 11 90 82 12 54 72 ...
##  $ Week3                : int  82 84 84 78 16 30 130 0 156 108 ...
##  $ Week4                : int  141 66 48 48 11 36 83 0 66 12 ...
##  $ Week5                : int  200 114 30 210 13 66 97 0 102 48 ...
##  $ Week6                : int  189 78 78 6 15 30 125 40 72 84 ...
##  $ Week7                : int  242 48 72 36 23 36 36 37 12 24 ...
##  $ Week8                : int  235 48 60 18 30 42 38 43 60 84 ...
##  $ Week9                : int  2 54 54 78 3 48 0 0 12 24 ...
##  $ Week10               : int  200 54 84 54 31 60 78 33 24 36 ...
##  $ Week11               : int  263 66 42 36 41 102 49 47 132 36 ...
##  $ Week12               : int  183 30 48 18 51 60 87 58 18 0 ...
##  $ Week13               : int  260 48 42 30 51 24 84 74 19 48 ...
##  $ Week14               : int  238 30 54 36 50 24 82 28 13 24 ...
##  $ Week15               : int  251 72 78 60 68 43 98 36 96 72 ...
##  $ Week16               : int  204 36 72 60 39 54 91 25 90 36 ...
##  $ Week17               : int  147 66 24 72 31 18 54 32 7 60 ...
##  $ Week18               : int  74 66 102 114 25 84 61 34 67 24 ...
##  $ Week19               : int  129 126 138 84 40 36 73 43 102 24 ...
##  $ Week20               : int  69 132 108 108 78 144 53 33 60 60 ...
##  $ Week21               : int  91 162 114 120 72 60 27 61 85 84 ...
##  $ Week22               : int  155 108 30 48 54 72 0 44 96 120 ...
##  $ Week23               : int  131 96 54 24 56 48 21 34 61 132 ...
##  $ Week24               : int  82 144 84 90 60 66 38 78 62 24 ...
##  $ Week25               : int  84 96 102 54 59 96 21 26 90 36 ...
##  $ Week26               : int  129 30 42 54 56 42 25 45 24 0 ...
##  $ Week27               : int  66 90 72 66 49 30 16 49 36 12 ...
##  $ Week28               : int  93 66 78 114 73 66 33 25 54 0 ...
##  $ Week29               : int  44 60 60 120 79 36 22 45 68 60 ...
##  $ Week30               : int  56 66 84 60 104 36 41 51 79 24 ...
##  $ Week31               : int  73 66 72 36 41 78 43 50 54 48 ...
##  $ Week32               : int  88 96 66 54 80 36 48 65 62 84 ...
##  $ Week33               : int  121 84 42 192 86 120 27 65 24 144 ...
##  $ Week34               : int  105 150 72 108 61 66 51 102 6 0 ...
##  $ Week35               : int  154 108 54 78 77 96 58 130 19 144 ...
##  $ Week36               : int  144 60 96 30 43 96 37 64 60 48 ...
##  $ Week37               : int  155 84 114 78 79 90 74 143 169 276 ...
##  $ Week38               : int  150 114 60 36 41 30 39 129 48 12 ...
##  $ Week39               : int  134 72 48 84 66 54 47 104 36 36 ...
##  $ Week40               : int  200 54 42 24 58 36 82 93 108 60 ...
##  $ Week41               : int  185 78 66 78 97 60 39 73 12 12 ...
##  $ Week42               : int  98 54 30 36 72 30 58 22 6 24 ...
##  $ Week43               : int  160 90 60 54 71 54 35 49 18 36 ...
##  $ Week44               : int  129 52 78 48 97 24 46 58 18 72 ...
##  $ Week45               : int  129 54 72 84 81 66 47 41 12 108 ...
##  $ Week46               : int  113 30 60 42 63 72 30 46 43 12 ...
##  $ Week47               : int  134 61 78 54 81 32 27 100 66 60 ...
##  $ Week48               : int  221 72 90 36 82 32 61 72 6 36 ...
##  $ Week49               : int  216 36 42 24 79 42 48 69 90 48 ...
##  $ Week50               : int  202 48 42 24 115 42 48 68 12 24 ...
##  $ Week51               : int  281 54 67 18 52 36 68 68 1 12 ...
##  $ Week52               : int  120 66 38 84 73 66 67 93 26 24 ...
##  $ Week53               : int  121 24 18 54 33 18 12 4 6 24 ...
##  $ Grand_Total          : int  7823 3913 3509 3416 3038 2941 2922 2880 2782 2750 ...
```

Time to export the data sets to CSV files so I can use them in my analysis.

```
# Exporting the datasets to CSV files
write.csv(High_Volume_Weekly_DS, "/Users/emilio/Downloads/High_Volume_Weekly_DS.csv", row.names = FALSE
write.csv(Medium_Volume_Weekly_DS, "/Users/emilio/Downloads/Medium_Volume_Weekly_DS.csv", row.names = F
write.csv(Low_Volume_Weekly_DS, "/Users/emilio/Downloads/Low_Volume_Weekly_DS.csv", row.names = FALSE)
```

Now time to work on the EDA!

My next goal for this project is to discover patterns with the data, and what I can find to help others know to understand. Particullarly, if I were able to find Moving Average Sales trends from any product or products, especially a top selling product, I can understand the basis of how my algorithm could work. My first goal will try to uncover…

- Moving Average Salaes Trends for a Top-Selling Product Working with Weekly data especially will help look at trends better.

I will choose the most sold amount of products across all stores, for ALL of 2024.

Since we have the "Grand_Total" Variable, all I have to do is look at the highest grand total for all the products and see which is the highst.

```
# Finding the top-selling product for 2024
top_selling_product_high_vol_store <- High_Volume_Weekly_DS |>
  arrange(desc(Grand_Total)) |>
  slice(1)

top_selling_product_medium_vol_store <- Medium_Volume_Weekly_DS |>
  arrange(desc(Grand_Total)) |>
  slice(1)

top_selling_product_low_vol_store <- Low_Volume_Weekly_DS |>
  arrange(desc(Grand_Total)) |>
  slice(1)

top_selling_product_high_vol_store
```

```
##   ItemID         Description Bottles_Per_Case Cost_Amount_Per_Bottle Total_Cost
## 1  18467 SCOTTY'S VODKA 50ML               96                   0.99      95.04
##   Week1 Week2 Week3 Week4 Week5 Week6 Week7 Week8 Week9 Week10 Week11 Week12
## 1    60    93    93    94    89   114   145    94    63    130     68    153
##   Week13 Week14 Week15 Week16 Week17 Week18 Week19 Week20 Week21 Week22 Week23
## 1     90     85     71    132    127    172     94    137    117    121     96
##   Week24 Week25 Week26 Week27 Week28 Week29 Week30 Week31 Week32 Week33 Week34
## 1    142    114    167    159    125    142    203    165    174    249    183
##   Week35 Week36 Week37 Week38 Week39 Week40 Week41 Week42 Week43 Week44 Week45
## 1    205    255    225    235    233    240    233    183    168    167    164
##   Week46 Week47 Week48 Week49 Week50 Week51 Week52 Week53 Grand_Total
## 1    170    159    137     99    118    163    102     76        7690
```

```
top_selling_product_medium_vol_store
```

```
##   ItemID                           Description Bottles_Per_Case
## 1  70417 FIREBALL CINN WHISKY 50ML/10PK LOOSE              120
##   Cost_Amount_Per_Bottle Total_Cost Week1 Week2 Week3 Week4 Week5 Week6 Week7
## 1                   1.09      130.8    76    60   137   136   115   127    81
##   Week8 Week9 Week10 Week11 Week12 Week13 Week14 Week15 Week16 Week17 Week18
## 1   113   118    158    101    128    160    102     86    117     91    121
##   Week19 Week20 Week21 Week22 Week23 Week24 Week25 Week26 Week27 Week28 Week29
## 1    109    102    110    100    110     84     55     72     73     76     80
##   Week30 Week31 Week32 Week33 Week34 Week35 Week36 Week37 Week38 Week39 Week40
## 1     57     79     61     82    131     87    103     97    113     97    138
##   Week41 Week42 Week43 Week44 Week45 Week46 Week47 Week48 Week49 Week50 Week51
## 1    126     90    112    154    119    135    109    105    155    131    137
##   Week52 Week53 Grand_Total
## 1    124     26        5687
```

```
top_selling_product_low_vol_store
```

```
##   ItemID                           Description Bottles_Per_Case
## 1  70417 FIREBALL CINN WHISKY 50ML/10PK LOOSE              120
##   Cost_Amount_Per_Bottle Total_Cost Week1 Week2 Week3 Week4 Week5 Week6 Week7
## 1                   1.09      130.8   141    88    82   141   200   189   242
##   Week8 Week9 Week10 Week11 Week12 Week13 Week14 Week15 Week16 Week17 Week18
## 1   235     2    200    263    183    260    238    251    204    147     74
##   Week19 Week20 Week21 Week22 Week23 Week24 Week25 Week26 Week27 Week28 Week29
## 1    129     69     91    155    131     82     84    129     66     93     44
##   Week30 Week31 Week32 Week33 Week34 Week35 Week36 Week37 Week38 Week39 Week40
## 1     56     73     88    121    105    154    144    155    150    134    200
##   Week41 Week42 Week43 Week44 Week45 Week46 Week47 Week48 Week49 Week50 Week51
## 1    185     98    160    129    129    113    134    221    216    202    281
##   Week52 Week53 Grand_Total
## 1    120    121        7823
```

As we see above, the top selling product for the.. High Volume store = "SCOTTY'S VODKA 50ML" Medium Volume store = "FIREBALL CINN WHISKEY 50ML/10PK LOOSE" Low Volume store = "FIREBALL CINN WHISKEY 50ML/10PK LOOSE".

Now for curiosity, I want to use the moving average across a set of weeks to see if I can find trends. I will have to use the zoo package for this.

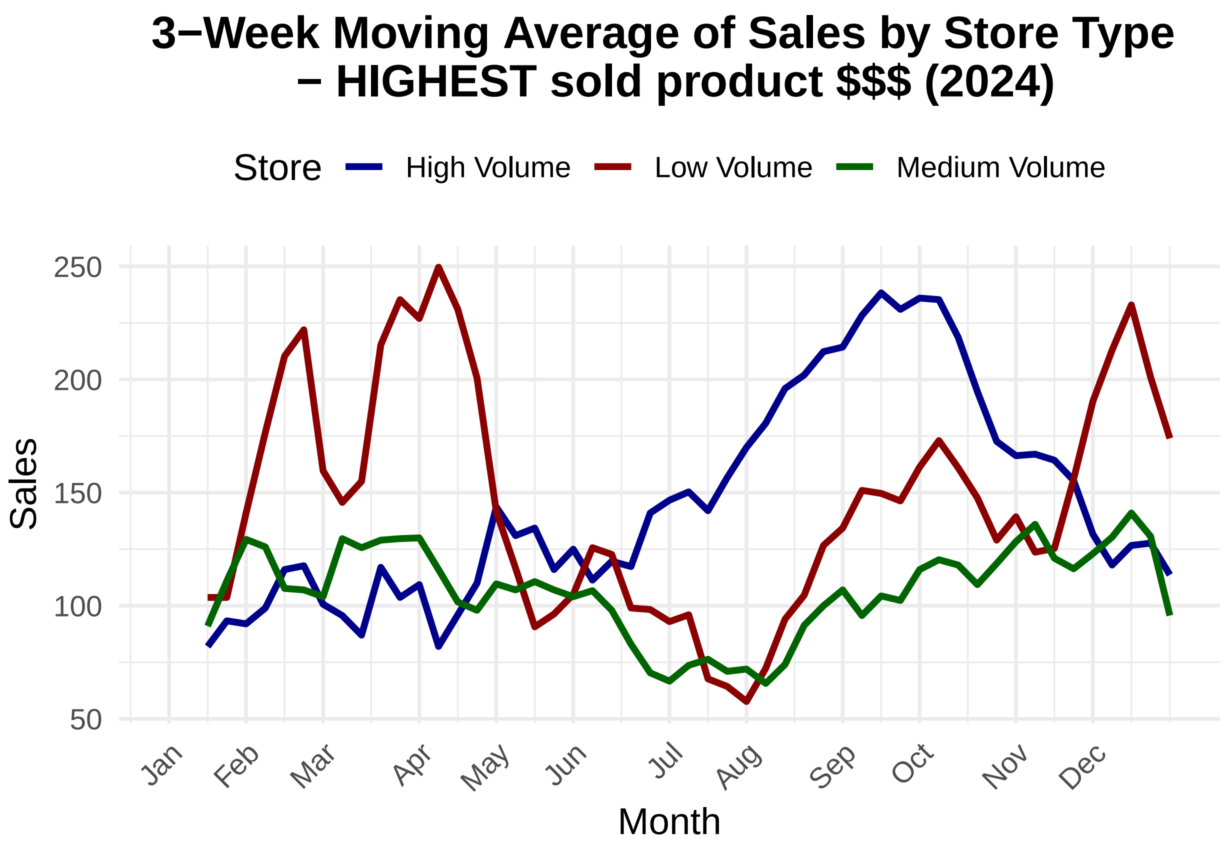
```
library(zoo) #Used Google to help understnad how to use this package
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
# Sales for 53 weeks in a vector
weekly_sales_high <- as.numeric(High_Volume_Weekly_DS[1, paste0("Week", 1:53)])
weekly_sales_medium <- as.numeric(Medium_Volume_Weekly_DS[1, paste0("Week", 1:53)])
weekly_sales_low <- as.numeric(Low_Volume_Weekly_DS[1, paste0("Week", 1:53)])
```

```r
# Will use a 4-week moving average
moving_avg_high <- rollmean(weekly_sales_high, k = 3, fill = NA, align = "right")
moving_avg_medium <- rollmean(weekly_sales_medium, k = 3, fill = NA, align = "right")
moving_avg_low <- rollmean(weekly_sales_low, k = 3, fill = NA, align = "right")

# Below I will put approximate starting week for each month to show when the months start
month_weeks <- c(1, 5, 9, 14, 18, 22, 27, 31, 36, 40, 45, 49)
month_labels <- c("Jan", "Feb", "Mar", "Apr", "May", "Jun",
                  "Jul", "Aug", "Sep", "Oct", "Nov", "Dec")

# Creating a data frame for plotting
moving_avg_df_high <- data.frame(
  Week = 1:53,
  Sales = moving_avg_high,
  Store = "High Volume"
)

moving_avg_df_medium <- data.frame(
  Week = 1:53,
  Sales = moving_avg_medium,
  Store = "Medium Volume"
)

moving_avg_df_low <- data.frame(
  Week = 1:53,
  Sales = moving_avg_low,
  Store = "Low Volume"
)

# Combining all data frames
moving_avg_df <- bind_rows(moving_avg_df_high, moving_avg_df_medium, moving_avg_df_low)

ggplot(moving_avg_df, aes(x = Week, y = Sales, color = Store)) +
  geom_line(size = 1.2, na.rm = TRUE) +
  scale_x_continuous(breaks = month_weeks, labels = month_labels) +
  scale_color_manual(values = c("High Volume" = "darkblue",
                                "Medium Volume" = "darkgreen",
                                "Low Volume" = "darkred")) +
  labs(
    title = "3-Week Moving Average of Sales by Store Type \n - HIGHEST sold product $$$ (2024)",
    x = "Month",
    y = "Sales",
    color = "Store"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    plot.title = element_text(face = "bold", hjust = 0.5),
    axis.text.x = element_text(angle = 45, hjust = 1),
    legend.position = "top"
  )
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```



Oh wow.. as you can see from the above, the 4-week Moving Averages for different volume of stores for the HIGHEST selling product all year long has been different for different weeks. Towards the end of the year, the sales for the high volume store have been increasing, while the sales for the medium and low volume stores have been decreasing in the middle of the weeks (as you can see... around July and August). This is very interesting to see. I will have to look at this more closely later on.

```r
#Me playing around with making the algorithm.

# Example weekly sales for the highest product (SCOTTY'S VODKA 50ML) as an example
period1 <- c(60, 93, 93, 94, 89)
period2 <- c(114, 145, 94, 63, 130)
period3 <- c(90, 95, 100, 98, 97)

# Calculating average sales for each period
avg1 <- mean(period1)
avg2 <- mean(period2)
avg3 <- mean(period3)

# Apply weights: P1 = 15%, P2 = 25%, P3 = 60%
weighted_avg <- (0.15 * avg1) + (0.25 * avg2) + (0.60 * avg3)
weighted_avg
```

```
## [1] 97.77
```

Okay simple. I split the first 3 periods (Each period if 5 weeks) from the first 15 weeks of the dataset (the start of january) and I calculated the average for each. Then I put calculated using weighted average.

```
reorder_threshold_days <- 10 # The number of days of sales you want to always have in stock before orde
lead_time_days <- 10

# Minimum Shelf Stock
MSS <- weighted_avg * reorder_threshold_days #The lowest amount of product the ABS stores should keep o
# MSS = (Average daily sales) × (Reorder Threshold)
# Exp: I need atleast # alcohol stores at all times. That's the backup stash, just in case demand sudde

#NOTE: Lead time is 10 days, so I will need to order more alcohol 10 days before I run out of stock.


# Reorder Quantity
reorder_qty <- (weighted_avg * lead_time_days) + MSS
# Reorder Quantity = (Average daily sales) × (Lead time) + MSS

reorder_qty
```

```
## [1] 1955.4
```

Cool! The reorder_qty is 1995.4. That means that I will need to order 1955.4 bottles of SCOTTY'S VODKA 50ML every time you reorder (which is every 10 days — the lead time).

In the future, I wil try to implement this algorithm to the rest of the products in the dataset. I will also try to implement this algorithm to the other two datasets (Medium and Low Volume stores) to see if I can find any patterns or trends, and make possibel visualizations with this. For now, I will keep how everything is.