Juweria Ali

2022-06-20

Loading required libraries

```
library(dplyr) #data wrangling

library(caret) #machine learning algorithms

library(stringr) #string manipulation
library(tidyr) #data manipulation
library(ggplot2) #data visualisation
library(lubridate) #date conversions
```

Loading the datasets

```
eventsdf <- read.csv("eventData.csv", header=T, stringsAsFactors=T)
weatherdf <- read.csv("weatherData.csv", header=T, stringsAsFactors=T)
```

Exploring the events dataset

```
eventsdf <- as_tibble(eventsdf) # to see datatypes along with data
glimpse(eventsdf) # makes it possible to see every column in the dataframe

## Rows: 500
## Columns: 10
## $ Date     <fct> 15/02/2023, 29/07/2022, 28/09/2021, 24/04/2022,
23/03/2021, 2~
## $ EventID  <fct> UID-1442799, UID-1112881, UID-3623146, UID-1999065, UID-
17657~
## $ Day      <fct> WD, WD, WD, WE, WE, WD, WE, WD, WD, PH, WE, WD, WD, WD,
WE, W~
## $ Visitors <fct> >2200, 1981, 1729, 4063, 2643, 1203, ~2800, 2456, 2095,
2970,~
## $ Hours    <fct> 4hr, 3.5hr, 2.5, 5.5, 4.5, 3.5, 2hr 45min, 4.5, 3, 4,
3.5, 4,~
## $ Advert   <fct> Yes, Yes, No, Yes, No, No, No, Yes, No, No, Yes, No, No,
No, ~
## $ Music    <fct> Yes, No, No, Yes, No, Yes, Yes, No, No, Yes, Yes, No, No,
No,~
## $ Sport    <fct> No, No, No, Yes, Yes, No, No, Yes, No, Yes, No, Yes, No,
No, ~
## $ Type     <fct> M, X, X, MS, S, M, M, S, X, MS, M, S, X, X, M, X, S, X,
M, X,~
## $ Sales    <dbl> 34117.23, 14261.71, 6137.98, 67481.74, 16187.11,
11049.92, 20~
```

Cleaning Visitors column

```
unique(eventsdf$Visitors)# to view all unique values in the column
```

```
##   [1] >2200     1981      1729      4063      2643      1203      ~2800    
##   [8] 2456      2095      2970      1500-1600 1723      1477      1633     
##  [15] 2666      1708      2759      1813      2115      2275      1791     
##  [22] 4101      4123      2237      1942      2212      4428      5124     
##  [29] 5184      1976      2688      2322      2637      1824      2490     
##  [36] 1959      ~3900     ~2400     4234      2513      2281      3673     
##  [43] 1986      1583      >1800     3871      1540      2600      2566     
##  [50] 2056      1635      2329      2546      3440      2903      2828     
##  [57] 1900-2000 1847      2188      3458      2469      3943      1888     
##  [64] 4090      1532      1554      2092      4153      2845      2132     
##  [71] 2626      1920      1494      4446      1800-1900 2291      1355     
##  [78] 2386      2898      2220      2982      2213      2100-2200 4163     
##  [85] ~2300     2066      3776      ~3000     1881      2430      2321     
##  [92] 1783      2709      2756      2477      3082      1876      3769     
##  [99] 1503      2058      1661      780       2569      3245      ~1900    
## [106] 2366      1314      1991      2600-2700 3428      2633      1890     
## [113] 2123      2007      2462      1727      1704      1912      2740     
## [120] 2364      3600-3700 3674      2447      2725      2135      3378     
## [127] 1618      3019      368       3211      2908      3284      1911     
## [134] 3665      3286      1877      3493      2138      4716      2114     
## [141] 2088      1248      ~2000     1297      2207      ~1500     2721     
## [148] 1878      2119      4550      2540      2000-2100 2777      2144     
## [155] 2516      2727      1971      3007      3363      2570      1003     
## [162] 2530      1868      3219      >2700     1467      3507      2388     
## [169] 5968      1846      3932      1700-1800 2412      1533      3038     
## [176] 2880      2491      2223      2899      1865      >4400     1688     
## [183] 2605      2706      3906      1895      1623      3127      2076     
## [190] 1718      3794      2925      3583      2319      2625      2848     
## [197] >3800     2204      2019      2071      2175      3205      >3000    
## [204] 2407      2200-2300 ~1200     5493      1845      2945      1825     
## [211] 2564      2385      1891      1715      1413      3256      2603     
## [218] 4070      3935      1493      2363      3250      2090      2699     
## [225] 3248      1631      1951      1678      2500-2600 ~3500     4498     
## [232] 2000      >2600     1948      1927      1725      2126      1934     
## [239] 1696      1492      3372      2026      ~3700     ~3100     3204     
## [246] 2287      1997      2002      2182      2900      1653      4564     
## [253] 2579      1947      2258      1953      2980      1535      2099     
## [260] 1905      1720      2455      1709      680       3116      2358     
## [267] 2800-2900 3592      2722      1779      2020      2492      2416     
## [274] 2938      2140      2635      2219      1134      2549      1142     
## [281] 2080      1702      2326      1563      2226      1834      1343     
## [288] 2384      2571      1316      ~4700     2584      719       1504     
## [295] 1735      2255      1157      1968      3571      2130      3001     
## [302] 2773      2032      2400-2500 2693      4438      1821      3281     
## [309] 2810      2707      2397      2585      1829      >2500     2075     
## [316] 3373      3893      2016      4810      2163      5020      3067     
## [323] 3631      2653      1291      ~3400     1832      3701      2065     
## [330] 2134      1946      3938      1341      2554      2423      1950
## [337] 2389      1668      2183      1523      1984      1018      2907
## [344] 5216      1769      1658      2341      1253      3221      3885
## [351] 1622      3526      1740      1580      2139      2179      3029
## [358] 1943      2873      1506      3112      2084      2041      1122
## [365] 1657      1643      2276      2543      2168      2304      1062
## [372] 2387      2305      4113      2442      2488      2465      2860
## [379] 1036      2206      1427      1439      1919      3441      2109
## [386] 1471      1521      2672      1875      ~1400     1979      2421
## [393] 3350      ~2100     1869      2142      1774      1690      3365
## [400] 2010      1603      1641      2324      2444      3763      1767
## [407] 3612      >3100     2296      2460      2619      ~1600     1862
## [414] 3704      >1600     2766      3014      2872      1822      4302
## [421] 1901      3406      3480      2994      4984      2730      1249
## [428] 2184      3547      846       2655      3947      1748      2261
## [435] 1990      1989      2409      2203      3030      2277      962
## [442] 3111      1241      2858      2151      2715      3610      1481
## [449] 1660      1531      2159      3442      3916
## 453 Levels: ~1200 ~1400 ~1500 ~1600 ~1900 ~2000 ~2100 ~2300 ~2400 ... 962
```

From the above results we can see instances that will require transformation. Values with >,~,-,and no string are filtered out into placeholders df1,df2,df3,df4 respectively.The below code is used to filter those specific rows using the filter() function and the grepl() function (Bobbitt 2020).

```
df1 <- eventsdf %>% filter(grepl('>', Visitors))
df2 <- eventsdf %>% filter(grepl('~', Visitors))
df3 <- eventsdf %>% filter(grepl('-', Visitors))
df4 <- eventsdf %>% filter(!grepl('>|~|-', Visitors))

str(df3)
```

```
## tibble [15 x 10] (S3: tbl_df/tbl/data.frame)
##  $ Date    : Factor w/ 500 levels "01/01/2022","01/01/2023",..: 160 267
194 464 212 216 335 343 189 449 ...
##  $ EventID : Factor w/ 500 levels "UID-1006339",..: 423 280 234 251 271
115 237 250 143 433 ...
##  $ Day     : Factor w/ 3 levels "PH","WD","WE": 3 2 2 2 1 3 1 2 1 3 ...
##  $ Visitors: Factor w/ 453 levels "~1200","~1400",..: 58 138 113 192 300
400 168 92 192 218 ...
##  $ Hours   : Factor w/ 61 levels "0.5","1.5","1.5 hours",..: 21 31 35 19
24 33 21 29 26 22 ...
##  $ Advert  : Factor w/ 3 levels "","No","Yes": 3 2 3 3 3 2 2 2 1 2 ...
##  $ Music   : Factor w/ 3 levels "","No","Yes": 3 3 2 2 3 2 3 2 3 3 ...
##  $ Sport   : Factor w/ 2 levels "No","Yes": 1 1 1 1 2 2 2 1 2 2 ...
##  $ Type    : Factor w/ 4 levels "M","MS","S","X": 1 1 4 4 2 3 2 4 2 2 ...
##  $ Sales   : num [1:15] 21217 11977 9844 10516 52357 ...
```

Now that we have all the rows separated we treat them appropriatley. Replacing the “>” symbol and “~” symbol with a blank and assuming the remainder value to be the value of that instance.

```
df1$Visitors<- str_replace(df1$Visitors,">","") # replaces > with a blank
df2$Visitors<- str_replace(df2$Visitors,"~","") #replaces ~ with a blank
```

Treating instances that have a range. First separating the lower limit and upper limit in two columns Col1 and Col2 and then calculating the average. Then assigning the average values, as values of these instances.Finally deleting Col1 and Col2 as they are no longer useful.

```
df3<- df3%>% separate(Visitors, into = c("Col1","Col2"), sep = "-", remove =
TRUE)
df3$Col1 <- as.numeric(df3$Col1)
df3$Col2 <- as.numeric(df3$Col2)
df3 <- df3 %>% mutate(Visitors=(Col1+Col2)/2) # Calculating average
df3$Col1 <- NULL
df3$Col2 <-NULL

dfA<-union(df1,df2)
str(dfA)

## tibble [35 x 10] (S3: tbl_df/tbl/data.frame)
##  $ Date    : Factor w/ 500 levels "01/01/2022","01/01/2023",..: 241 188
279 192 436 310 346 198 486 105 ...
##  $ EventID : Factor w/ 500 levels "UID-1006339",..: 53 141 389 144 441 19
11 79 187 334 ...
##  $ Day     : Factor w/ 3 levels "PH","WD","WE": 2 1 1 3 1 3 2 2 3 2 ...
##  $ Visitors: chr [1:35] "2200" "1800" "2700" "4400" ...
##  $ Hours   : Factor w/ 61 levels "0.5","1.5","1.5 hours",..: 37 23 39 48
44 36 13 31 44 26 ...
##  $ Advert  : Factor w/ 3 levels "","No","Yes": 3 2 3 3 2 3 2 3 3 2 ...
##  $ Music   : Factor w/ 3 levels "","No","Yes": 3 2 3 2 2 3 2 2 2 2 ...
##  $ Sport   : Factor w/ 2 levels "No","Yes": 1 2 2 1 2 2 1 2 2 1 ...
##  $ Type    : Factor w/ 4 levels "M","MS","S","X": 1 3 2 4 3 2 4 3 3 4 ...
##  $ Sales   : num [1:35] 34117 8298 48551 29316 17786 ...

dfA$Visitors <- as.numeric(dfA$Visitors)

str(df4)

## tibble [450 x 10] (S3: tbl_df/tbl/data.frame)
##  $ Date    : Factor w/ 500 levels "01/01/2022","01/01/2023",..: 473 463
391 370 477 434 14 332 350 240 ...
##  $ EventID : Factor w/ 500 levels "UID-1006339",..: 17 335 128 95 26 83
107 488 41 408 ...
##  $ Day     : Factor w/ 3 levels "PH","WD","WE": 2 2 3 3 2 2 2 1 2 2 ...
##  $ Visitors: Factor w/ 453 levels "~1200","~1400",..: 160 103 424 309 36
272 190 351 100 53 ...
##  $ Hours   : Factor w/ 61 levels "0.5","1.5","1.5 hours",..: 23 9 45 33 21
33 19 31 31 9 ...
```

```
##  $ Advert   : Factor w/ 3 levels "","No","Yes": 3 2 3 2 2 3 2 2 2 2 ...
##  $ Music    : Factor w/ 3 levels "","No","Yes": 2 2 3 2 3 2 2 3 2 2 ...
##  $ Sport    : Factor w/ 2 levels "No","Yes": 1 1 2 2 1 2 1 2 2 1 ...
##  $ Type     : Factor w/ 4 levels "M","MS","S","X": 4 4 2 3 1 3 4 2 3 4 ...
##  $ Sales    : num [1:450] 14262 6138 67482 16187 11050 ...

df4$Visitors <- as.numeric(df4$Visitors)
dfB<-union(df3,df4)

eventsdf<-union(dfA,dfB)
```

Cleaning Hours column

A similar approach to cleaning the visitors column has been adapted below

```
unique(eventsdf$Hours)

##  [1] 4hr       3.5hr     4hr 0min  5.5Hr     5 hours   4.5Hr     2hr
##  [8] 4         3Hr       4.5       5hr 0min  2hr 45min 5         5hr 15min
## [15] 3hr 0min  4hr 30min 3hr 15min 2.5 hours 2.5       5.5       5hr
## [22] 4hr 15min 3.5       4.5hr     3         3.5Hr     3hr 30min 3.5 hours
## [29] 4Hr       2 hours   6Hr       4 hours   3hr       4.5 hours 2.5hr
## [36] 5.5 hours 6.5       4hr 45min 3 hours   6         7hr       5.5hr
## [43] 2hr 15min 5Hr       3hr 45min 2Hr       6hr 15min 1.5 hours 6.5Hr
## [50] 2hr 30min 2         5hr 45min 2hr 0min  5hr 30min 1.5       0.5
## [57] 2.5Hr     1hr 30min 1.5hr     1hr 45min 6 hours
## 61 Levels: 0.5 1.5 1.5 hours 1.5hr 1hr 30min 1hr 45min 2 2 hours ... 7hr

df1 <- eventsdf %>% filter(grepl('min', Hours))
df2<- eventsdf %>% filter(grepl('hr|Hr|Hours', Hours))%>%
filter(!grepl('min', Hours))
df3 <- eventsdf %>% filter(!grepl('min|hr|Hr|Hours', Hours))

df1<- df1%>% separate(Hours, into = c("Col1","Col2"), sep = " ", remove =
TRUE)
df1$Col1<- str_replace(df1$Col1,"hr","")
df1$Col2<- str_replace(df1$Col2,"min","")
str(df1)

## tibble [75 x 11] (S3: tbl_df/tbl/data.frame)
##  $ Date     : Factor w/ 500 levels "01/01/2022","01/01/2023",..: 279 171
498 398 158 243 322 178 135 202 ...
##  $ EventID  : Factor w/ 500 levels "UID-1006339",..: 389 65 193 384 385 345
348 288 140 172 ...
##  $ Day      : Factor w/ 3 levels "PH","WD","WE": 1 2 3 3 2 2 3 1 3 2 ...
##  $ Visitors: num [1:75] 2700 1800 2800 3000 1900 2000 1500 3100 2300 2100
...
##  $ Col1     : chr [1:75] "4" "5" "2" "5" ...
##  $ Col2     : chr [1:75] "0" "0" "45" "15" ...
##  $ Advert   : Factor w/ 3 levels "","No","Yes": 3 2 2 2 2 2 3 3 2 3 ...
##  $ Music    : Factor w/ 3 levels "","No","Yes": 3 2 3 3 2 2 3 3 3 2 ...
##  $ Sport    : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 1 2 2 1 1 ...
##  $ Type    : Factor w/ 4 levels "M","MS","S","X": 2 4 1 1 4 4 2 2 1 4 ...
##  $ Sales   : num [1:75] 48551 7312 20300 12834 7301 ...
```

```
df1$Col1 <- as.numeric(df1$Col1)
df1$Col2 <- as.numeric(df1$Col2)
df1 <- df1 %>% mutate(Hours=((Col2/60)+Col1)) # Converting minutes to hours
str(df1)
```

```
## tibble [75 x 12] (S3: tbl_df/tbl/data.frame)
##  $ Date    : Factor w/ 500 levels "01/01/2022","01/01/2023",..: 279 171
498 398 158 243 322 178 135 202 ...
##  $ EventID : Factor w/ 500 levels "UID-1006339",..: 389 65 193 384 385 345
348 288 140 172 ...
##  $ Day     : Factor w/ 3 levels "PH","WD","WE": 1 2 3 3 2 2 3 1 3 2 ...
##  $ Visitors: num [1:75] 2700 1800 2800 3000 1900 2000 1500 3100 2300 2100
...
##  $ Col1    : num [1:75] 4 5 2 5 3 4 3 5 3 4 ...
##  $ Col2    : num [1:75] 0 0 45 15 0 30 15 15 0 15 ...
##  $ Advert  : Factor w/ 3 levels "","No","Yes": 3 2 2 2 2 2 3 3 2 3 ...
##  $ Music   : Factor w/ 3 levels "","No","Yes": 3 2 3 3 2 2 3 3 3 2 ...
##  $ Sport   : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 1 2 2 1 1 ...
##  $ Type    : Factor w/ 4 levels "M","MS","S","X": 2 4 1 1 4 4 2 2 1 4 ...
##  $ Sales   : num [1:75] 48551 7312 20300 12834 7301 ...
##  $ Hours   : num [1:75] 4 5 2.75 5.25 3 4.5 3.25 5.25 3 4.25 ...
```

```
df1$Col1 <- NULL
df1$Col2 <- NULL
```

```
df2$Hours<- str_replace(df2$Hours,'hr|Hr|Hours',"")
str(df2)
```

```
## tibble [150 x 10] (S3: tbl_df/tbl/data.frame)
##  $ Date    : Factor w/ 500 levels "01/01/2022","01/01/2023",..: 241 188
192 310 346 105 111 255 257 497 ...
##  $ EventID : Factor w/ 500 levels "UID-1006339",..: 53 141 144 19 11 334
13 346 305 85 ...
##  $ Day     : Factor w/ 3 levels "PH","WD","WE": 2 1 3 3 2 2 2 2 2 2 ...
##  $ Visitors: num [1:150] 2200 1800 4400 3000 2600 2500 1600 2300 2300 1600
...
##  $ Hours   : chr [1:150] "4" "3.5" "5.5" "4.5" ...
##  $ Advert  : Factor w/ 3 levels "","No","Yes": 3 2 3 3 2 2 3 2 2 2 ...
##  $ Music   : Factor w/ 3 levels "","No","Yes": 3 2 2 3 2 2 3 2 2 2 ...
##  $ Sport   : Factor w/ 2 levels "No","Yes": 1 2 1 2 1 1 1 1 1 2 ...
##  $ Type    : Factor w/ 4 levels "M","MS","S","X": 1 3 4 2 4 4 1 4 4 3 ...
##  $ Sales   : num [1:150] 34117 8298 29316 60027 10956 ...
```

```
df2$Hours <- as.numeric(df2$Hours)
str(df2)
```

```
## tibble [150 x 10] (S3: tbl_df/tbl/data.frame)
##  $ Date    : Factor w/ 500 levels "01/01/2022","01/01/2023",..: 241 188
```

```
192 310 346 105 111 255 257 497 ...
##  $ EventID : Factor w/ 500 levels "UID-1006339",..: 53 141 144 19 11 334
13 346 305 85 ...
##  $ Day     : Factor w/ 3 levels "PH","WD","WE": 2 1 3 3 2 2 2 2 2 2 ...
##  $ Visitors: num [1:150] 2200 1800 4400 3000 2600 2500 1600 2300 2300 1600
...
##  $ Hours   : num [1:150] 4 3.5 5.5 4.5 2 3 3 4.5 5 3.5 ...
##  $ Advert  : Factor w/ 3 levels "","No","Yes": 3 2 3 3 2 2 3 2 2 2 ...
##  $ Music   : Factor w/ 3 levels "","No","Yes": 3 2 2 3 2 2 3 2 2 2 ...
##  $ Sport   : Factor w/ 2 levels "No","Yes": 1 2 1 2 1 1 1 1 1 2 ...
##  $ Type    : Factor w/ 4 levels "M","MS","S","X": 1 3 4 2 4 4 1 4 4 3 ...
##  $ Sales   : num [1:150] 34117 8298 29316 60027 10956 ...
```

```
dfA <- union(df1,df2)
```

```
str(df3)
```

```
## tibble [275 x 10] (S3: tbl_df/tbl/data.frame)
##  $ Date    : Factor w/ 500 levels "01/01/2022","01/01/2023",..: 436 198
486 86 290 80 302 25 149 264 ...
##  $ EventID : Factor w/ 500 levels "UID-1006339",..: 441 79 187 332 188 337
244 450 43 208 ...
##  $ Day     : Factor w/ 3 levels "PH","WD","WE": 1 2 3 3 3 1 2 3 3 3 ...
##  $ Visitors: num [1:275] 3800 1800 2700 3100 3900 2400 1200 3500 3700 4700
...
##  $ Hours   : Factor w/ 61 levels "0.5","1.5","1.5 hours",..: 44 31 44 33
43 33 10 33 9 31 ...
##  $ Advert  : Factor w/ 3 levels "","No","Yes": 2 3 3 3 3 3 2 3 2 2 ...
##  $ Music   : Factor w/ 3 levels "","No","Yes": 2 2 2 2 3 3 2 3 2 2 ...
##  $ Sport   : Factor w/ 2 levels "No","Yes": 2 2 2 1 1 2 1 1 1 1 ...
##  $ Type    : Factor w/ 4 levels "M","MS","S","X": 3 3 3 4 1 2 4 1 4 4 ...
##  $ Sales   : num [1:275] 17786 28797 22198 25523 59184 ...
```

```
str(dfA)
```

```
## tibble [225 x 10] (S3: tbl_df/tbl/data.frame)
##  $ Date    : Factor w/ 500 levels "01/01/2022","01/01/2023",..: 279 171
498 398 158 243 322 178 135 202 ...
##  $ EventID : Factor w/ 500 levels "UID-1006339",..: 389 65 193 384 385 345
348 288 140 172 ...
##  $ Day     : Factor w/ 3 levels "PH","WD","WE": 1 2 3 3 2 2 3 1 3 2 ...
##  $ Visitors: num [1:225] 2700 1800 2800 3000 1900 2000 1500 3100 2300 2100
...
##  $ Advert  : Factor w/ 3 levels "","No","Yes": 3 2 2 2 2 2 3 3 2 3 ...
##  $ Music   : Factor w/ 3 levels "","No","Yes": 3 2 3 3 2 2 3 3 3 2 ...
##  $ Sport   : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 1 2 2 1 1 ...
##  $ Type    : Factor w/ 4 levels "M","MS","S","X": 2 4 1 1 4 4 2 2 1 4 ...
##  $ Sales   : num [1:225] 48551 7312 20300 12834 7301 ...
##  $ Hours   : num [1:225] 4 5 2.75 5.25 3 4.5 3.25 5.25 3 4.25 ...
```

```
df3$Hours <- as.numeric(df3$Hours)
eventsdf <- union(dfA,df3)
```

Treating missing values in Advert & Music column by filtering them out as they are very few in number

```
eventsdf <- eventsdf %>% filter(!Advert =="") %>% filter(!Music =="")
```

Exploring the weather dataset

```
weatherdf <- as_tibble(weatherdf) # to see datatypes along with data
glimpse(weatherdf) # makes possible to see every column in the dataframe

## Rows: 500
## Columns: 6
## $ Date    <fct> 15/02/2023, 29/07/2022, 28/09/2021, 24/04/2022,
23/03/2021, 29~
## $ Temp    <dbl> 22.9, 18.1, 11.6, 16.0, 6.3, 13.4, 18.7, 22.4, 11.0, 3.0,
8.0,~
## $ Rain    <dbl> 0.8, 0.2, 7.8, 0.7, 5.5, -2.9, 1.5, 1.5, 0.7, 1.0, 9.1,
6.4, 1~
## $ Wind    <dbl> 0.7, 9.6, 4.5, 5.2, 10.8, 1.4, 3.0, 7.6, 4.0, 4.9, 8.0,
6.8, 7~
## $ WindDir <fct> E, W, E, S, E, N, N, S, S, S, S, E, S, S, W, W, E, N, W,
S, S,~
## $ SnowIce <fct> No, No, No, No, No, No, No, No, No, Snow, No, No, No, No,
, No~
```

Replacing row value “neither” with “No” in SnowIce column

As both values ‘neither’ and ‘No’ imply the same we replace instances with value ‘neither’(as these are few in number comparitively) with value ‘No’.This also ensures uniformity of the column.

```
weatherdf$SnowIce[weatherdf$SnowIce == "neither"] <- "No"
```

Treating missing values in SnowIce column.

```
weatherdf %>% filter(SnowIce == "") #Filtering out instances with missing
values

## # A tibble: 22 x 6
##    Date        Temp  Rain  Wind WindDir SnowIce
##    <fct>      <dbl> <dbl> <dbl> <fct>   <fct>
##  1 05/09/2021  11.1   1.4   3.4 W       ""
##  2 02/07/2023  31.7   1     4.5 E       ""
##  3 02/04/2022  15.5   2.8   7.3 E       ""
##  4 16/07/2023  36.7   0.7   6   E       ""
##  5 03/09/2022  18.7   0.3  10.8 W       ""
##  6 04/02/2023  22.5   1.9   8.8 N       ""
##  7 24/05/2022  16.7   4.9   5   E       ""
##  8 10/11/2021  12.4   1    10.9 E       ""
##  9 23/09/2021  11.5   4.9  10.9 N       ""
## 10 17/12/2022  21.1   0.3   5.3 N       ""
## # ... with 12 more rows
```

By looking at the results above it is noticed that there are 22 instances with missing values. It is general knowledge that snow and ice occur at low temperatures. We can see from the figures above that all the instances with blank or missing values have corresponding temperature values, lowest being 10.3 and highest is 36.7. These temperatures are not suitable for snow or ice. Hence, we make an assumption that the missing values are a “No” and replace them with the same below.

```
weatherdf$SnowIce[weatherdf$SnowIce == ""] <- "No"
```

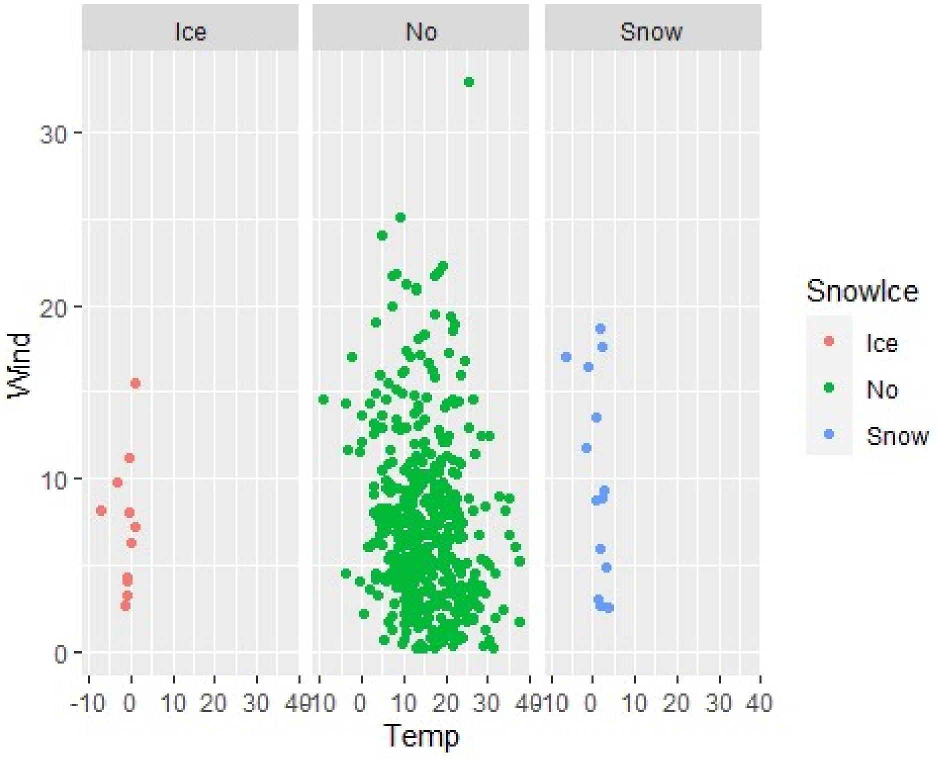
Task 2 Merging the datasets

```
alldata <- merge(x = eventsdf, y = weatherdf,
                        by.x = c("Date"),
                        by.y=c("Date"), all.x = FALSE, all.y=TRUE)
write.csv(alldata, "alldata.csv")
```

Task 3 Exploratory Data analysis

Chart 1

```
ggplot(alldata,aes(Temp,Wind,color=SnowIce))+ geom_point()+
facet_wrap(vars(SnowIce))
```

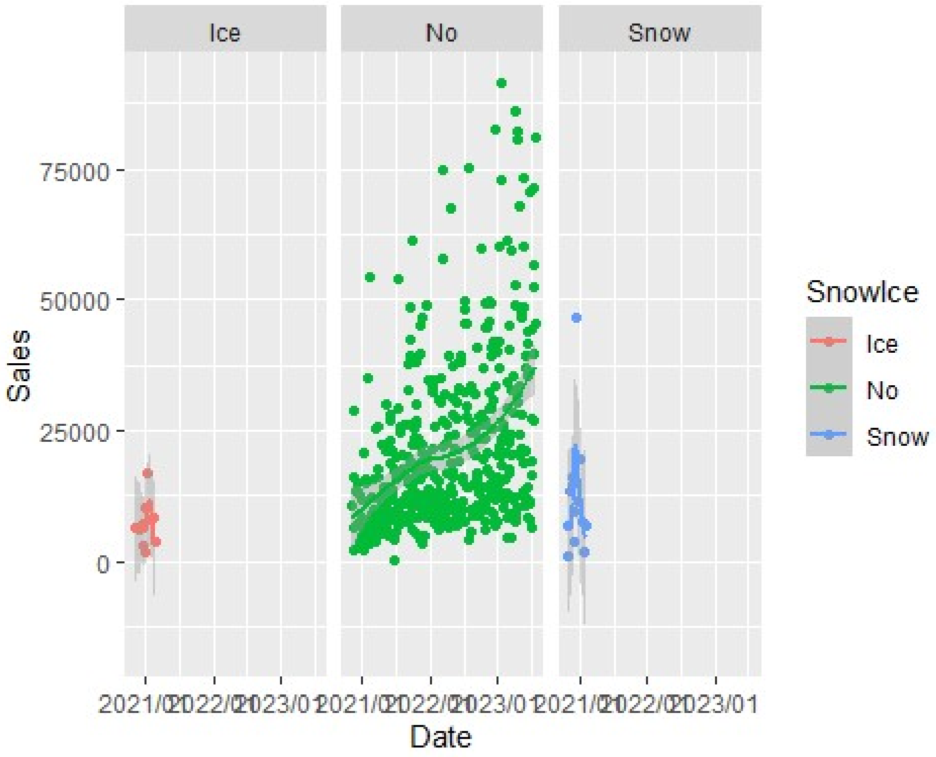


Discussion :

We can see from the graph above that ice forms at low temperatures, typically between -8 and 0 degrees Celsius, and low wind speeds, between 3 and 15.5 kilometres per hour. Temperatures between -1-5 degrees Celsius, as well as winds of 2–19 km/hr, result in snowfall. We also notice that compared to situations when there was no snow or ice, comparably few instances have happened in these circumstances. We may say that majority of the instances have occured when there is neither snow nor ice.

Chart 2

```
str(alldata)

## 'data.frame':    500 obs. of  15 variables:
##  $ Date    : Factor w/ 500 levels "01/01/2022","01/01/2023",..: 1 2 3 4 5
6 7 8 9 10 ...
##  $ EventID : Factor w/ 500 levels "UID-1006339",..: 330 296 185 NA 476 224
401 404 228 166 ...
##  $ Day     : Factor w/ 3 levels "PH","WD","WE": 2 2 2 NA 2 2 2 1 3 2 ...
##  $ Visitors: num  271 292 376 NA 49 105 55 329 274 322 ...
##  $ Advert  : Factor w/ 3 levels "","No","Yes": 2 3 3 NA 3 3 2 2 3 2 ...
##  $ Music   : Factor w/ 3 levels "","No","Yes": 2 2 2 NA 2 2 3 3 2 2 ...
##  $ Sport   : Factor w/ 2 levels "No","Yes": 1 1 1 NA 1 1 1 1 1 1 ...
##  $ Type    : Factor w/ 4 levels "M","MS","S","X": 4 4 4 NA 4 4 1 1 4 4 ...
##  $ Sales   : num  9070 20222 16858 NA 9674 ...
##  $ Hours   : num  10 33 33 NA 9 33 22 31 3.5 4.5 ...
##  $ Temp    : num  13.5 21.5 23.1 6.7 15.4 7.4 16.1 25.6 8.7 27.1 ...
##  $ Rain    : num  0.6 4.6 0.1 2.8 5.9 5.6 0.2 1 6.4 0.3 ...
##  $ Wind    : num  18.1 14.6 3.5 4.5 14.7 5.2 3.6 8.8 8.3 3.4 ...
##  $ WindDir : Factor w/ 4 levels "E","N","S","W": 2 3 3 2 1 4 3 3 4 1 ...
##  $ SnowIce : Factor w/ 5 levels "","Ice","neither",..: 4 4 4 4 4 4 4 4 4 4
...

alldata$Date <- dmy(alldata$Date)

ggplot(alldata,aes(Date,Sales,color=SnowIce))+ geom_point()+
facet_wrap(vars(SnowIce))+scale_x_date(date_labels = "%Y/%m")+geom_smooth()

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

## Warning: Removed 9 rows containing non-finite values (stat_smooth).

## Warning: Removed 9 rows containing missing values (geom_point).
```



Discussion:

In the chart above, we can see a trend in each of the three categories—ice, snow, and no. End of November 2020 to beginning of February 2021 sees the most snow and ice. After that, there is no longer any snow or ice in the following years. 2021 marked the peak for snow and ice before declining. It's interesting to note that sales when there was no snow or ice outnumber those when there was.

```
alldata$Sales <- as.integer(as.character(alldata$Sales))
bySnowIce <- group_by(alldata, SnowIce)
groupedDetails <- summarise(bySnowIce,
                    count = n(),
                    averageSales = mean(Sales, na.rm=T),
                    medianSales = median(Sales, na.rm=T),
                    highestSales = max(Sales, na.rm=T),

                    )
groupedDetails

## # A tibble: 3 x 5
##   SnowIce count averageSales medianSales highestSales
##   <fct>   <int>        <dbl>       <dbl>        <int>
## 1 Ice        11        7176.       6656         16703
## 2 No        475       21121.      15686         91613
## 3 Snow       14       12549.      10460.        46524
```

Correlation between type of music and sales

```
alldata$Sales <- as.integer(as.character(alldata$Sales))
byMusic <- group_by(alldata, Music)
groupedDetails <- summarise(byMusic,
                    count = n(),
                    averageSales = mean(Sales, na.rm=T),
                    medianSales = median(Sales, na.rm=T),
                    highestSales = max(Sales, na.rm=T),

                    )
```

```
## Warning in max(Sales, na.rm = T): no non-missing arguments to max;
returning
## -Inf
```

```
groupedDetails
```

```
## # A tibble: 3 x 5
##   Music count averageSales medianSales highestSales
##   <fct> <int>        <dbl>       <dbl>        <dbl>
## 1 No      295       13514.       10696        85939
## 2 Yes     196       31174.       26557        91613
## 3 <NA>      9         NaN           NA         -Inf
```

Correlation between sport and sales

```
alldata$Sales <- as.integer(as.character(alldata$Sales))
bySport <- group_by(alldata, Sport)
groupedDetails <- summarise(bySport,
                    count = n(),
                    averageSales = mean(Sales, na.rm=T),
                    medianSales = median(Sales, na.rm=T),
                    highestSales = max(Sales, na.rm=T),

                    )
```

```
## Warning in max(Sales, na.rm = T): no non-missing arguments to max;
returning
## -Inf
```

```
groupedDetails
```

```
## # A tibble: 3 x 5
##   Sport count averageSales medianSales highestSales
##   <fct> <int>        <dbl>       <dbl>        <dbl>
## 1 No      341       14543.       11320        59184
## 2 Yes     150       34251.       29683        91613
## 3 <NA>      9         NaN           NA         -Inf
```

Correlation between Advert and sales

```
alldata$Sales <- as.integer(as.character(alldata$Sales))
byAdvert <- group_by(alldata, Advert)
```

```
groupedDetails <- summarise(byAdvert,
                    count = n(),
                    averageSales = mean(Sales, na.rm=T),
                    medianSales = median(Sales, na.rm=T),
                    highestSales = max(Sales, na.rm=T),

                    )

## Warning in max(Sales, na.rm = T): no non-missing arguments to max;
returning
## -Inf

groupedDetails

## # A tibble: 3 x 5
##   Advert count averageSales medianSales highestSales
##   <fct>  <int>        <dbl>       <dbl>        <dbl>
## 1 No       296       13273.       10812        49629
## 2 Yes      195       31631.       27092        91613
## 3 <NA>       9         NaN           NA         -Inf
```

Correlation between type of day and sales

```
alldata$Sales <- as.integer(as.character(alldata$Sales))
byDay <- group_by(alldata, Day)
groupedDetails <- summarise(byDay,
                    count = n(),
                    averageSales = mean(Sales, na.rm=T),
                    medianSales = median(Sales, na.rm=T),
                    highestSales = max(Sales, na.rm=T),

                    )

## Warning in max(Sales, na.rm = T): no non-missing arguments to max;
returning
## -Inf

groupedDetails

## # A tibble: 4 x 5
##   Day   count averageSales medianSales highestSales
##   <fct> <int>        <dbl>       <dbl>        <dbl>
## 1 PH       80       33183.      29086.        91613
## 2 WD      242       13461.      11148.        47636
## 3 WE      169       24762.      20330         82267
## 4 <NA>      9         NaN          NA          -Inf
```

Discussion:

We can plainly see how these factors affect sales numbers from the four correlation tables above between advertisement and sales, music and sales, sports and sales, and day and sales.

We can see that when the event was marketed, had music and sporting events, the sales were far higher than when these things weren't there. Similarly, the sales on public holidays were the highest, followed by weekends and the lowest on weekdays.

Preparing the dataset for learning

Removing columns that are not useful for learning as they do not add value to any further analysis

```
alldata <- subset(alldata, select = -c(EventID,WindDir))
```

Removing NA's

```
alldata <- na.omit(alldata)
```

The data from alldata is now ready for learning. If more data preparation is necessary, it can be done in accordance with the learning task that will be carried out.

References

- Bobbitt,Z., 2020. How to filter rows that contain a certain string using dplyr. [online].
  Torrance: Statology. Available from:
  https://www.statology.org/filter-rows-that-contain-string-dplyr/ [Accessed 26/06/2022].

- Ines, A., 2022. CMM535. [Recorded lecture week 1-5]. CMM 535 Data Science Development. School of Computing. The Robert Gordon University [Accessed 27/06/2022].