Mooc

Me Myself

2023-05-03

Opening packages for use

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     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 errors
```

```
library(lubridate)
```

Opening file

```
df <- read.csv("oFf9xfM_S82X_cXzP0vNNg_79b3b2a4a5364c45b80e9ccbc1bb4ef1_mod4_peer_rev_data.csv", strings
view(df)
str(df)
```

```
## 'data.frame':    34432 obs. of  6 variables:
##  $ Date        : chr  "1/14/2016" "7/2/2016" "7/9/2014" "9/10/2014" ...
##  $ Department  : chr  "Kabobs" "Sides" "Sides" "Sides" ...
##  $ Category    : chr  "Pork" "Rice" "Rice" "Rice" ...
##  $ CustomerCode: chr  "CWM11331L8O" "CWM11331L8O" "CXP4593H7E" "CWM11331L8O" ...
##  $ Price       : int  28 9 9 9 25 18 26 12 9 12 ...
##  $ Quantity    : int  11 5 14 6 7 13 9 6 11 22 ...
```

Step 1: Transforming Data

```
df$CustomerCode<-as.character(df$CustomerCode)
df$Department<-as.factor(df$Department)
df$Category<-as.factor(df$Category)
df$Date<-mdy(df$Date)
str(df)
```

```
## 'data.frame':    34432 obs. of  6 variables:
##  $ Date        : Date, format: "2016-01-14" "2016-07-02" ...
##  $ Department  : Factor w/ 3 levels "Entrees","Kabobs",..: 2 3 3 3 2 1 1 3 3 1 ...
##  $ Category    : Factor w/ 10 levels "Beef","Beef and Broccoli",..: 7 8 8 8 1 6 2 10 8 6 ...
##  $ CustomerCode: chr  "CWM11331L8O" "CWM11331L8O" "CXP4593H7E" "CWM11331L8O" ...
##  $ Price       : int  28 9 9 9 25 18 26 12 9 12 ...
##  $ Quantity    : int  11 5 14 6 7 13 9 6 11 22 ...
```

Step 2: Displaying summaries

```
summary(df$Quantity)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    1.00    8.00   11.00   11.31   15.00   24.00       7
```

```
summary(df$Price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    3.00   12.00   25.00   22.81   33.00   50.00      10
```

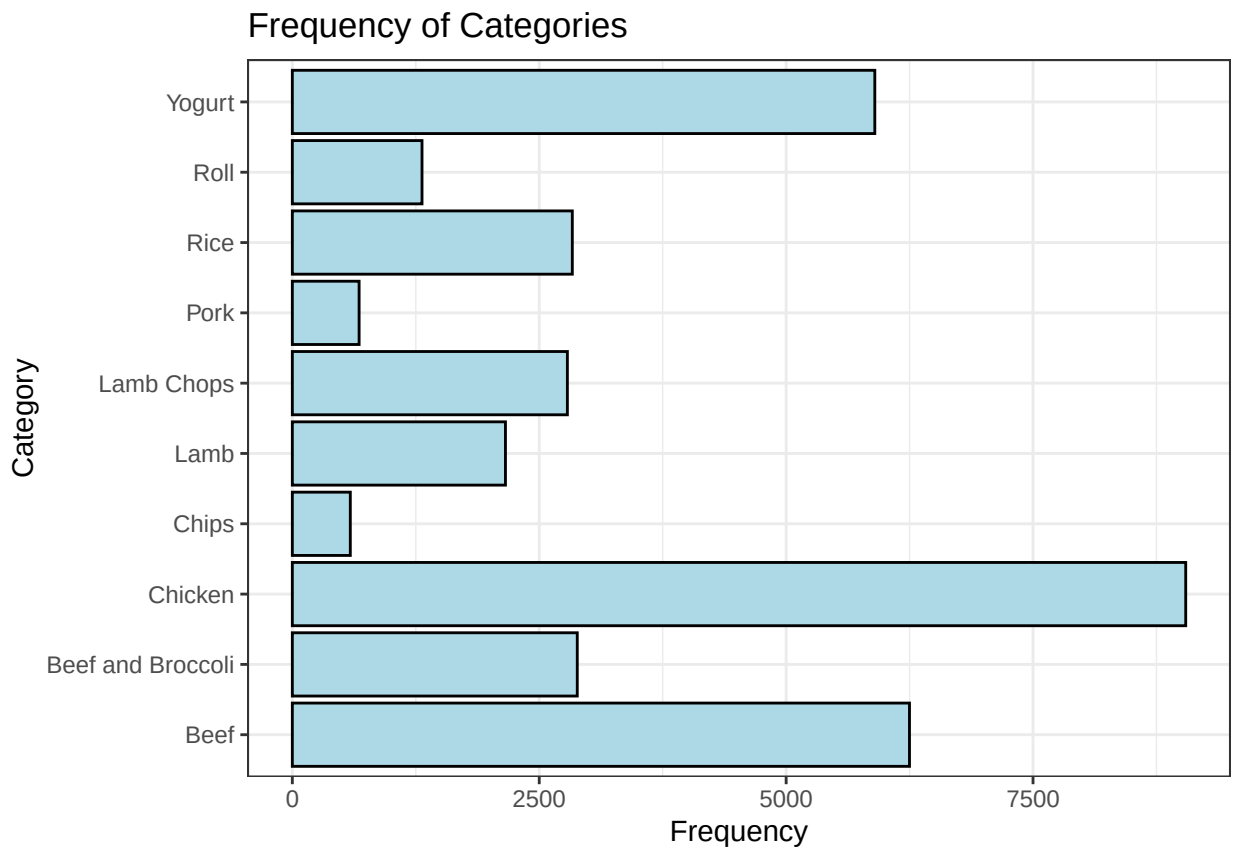
Step 3: Displaying NA count

```
sapply(df, function(x) sum(is.na(x)))
```

```
##         Date   Department     Category CustomerCode        Price     Quantity
##            0            0            0            0           10            7
```

Step 4: Displaying a bar chart

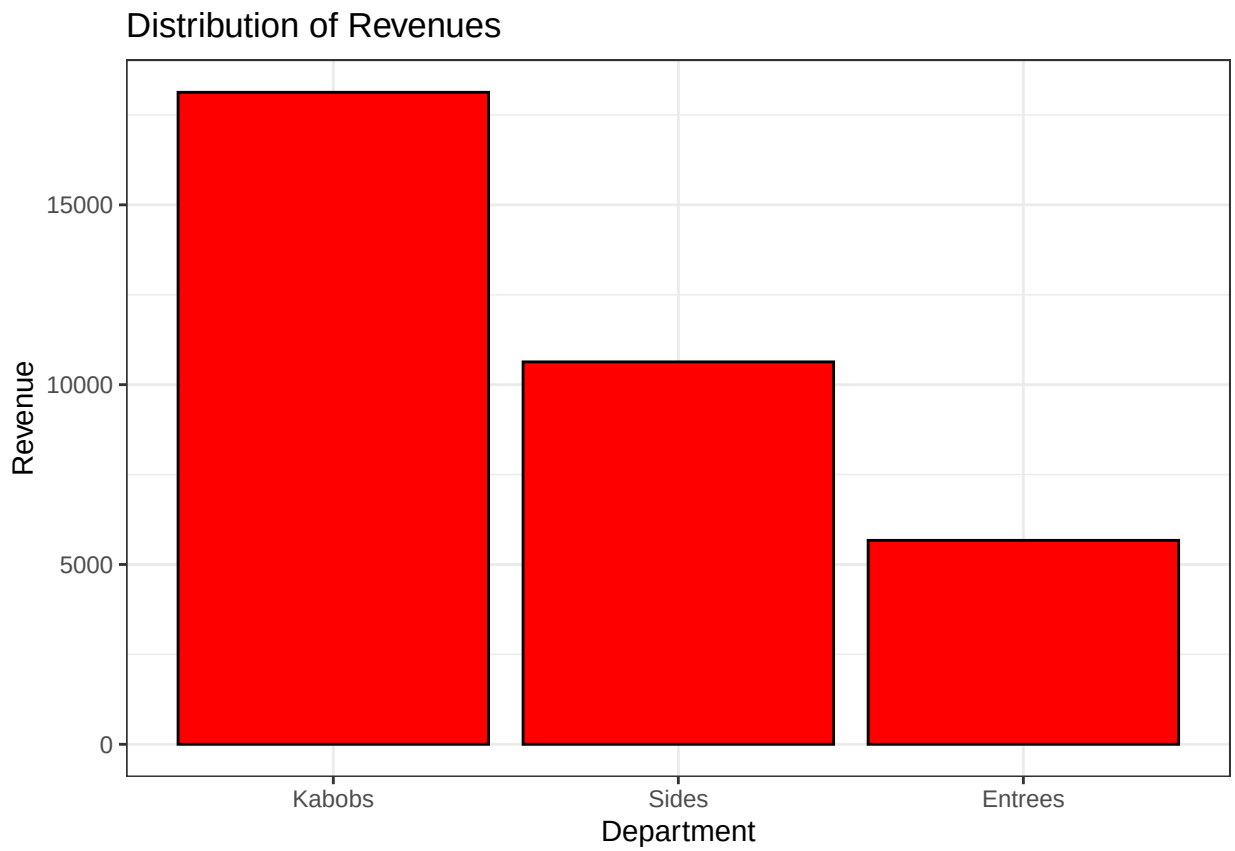
```
ggplot(df, aes(x = Category)) +
    geom_bar(fill = "lightblue", colour = "black") +
    labs(
      title = "Frequency of Categories",
      y = "Frequency",
      x = "Category"
      ) +
    theme_bw()+
    coord_flip()
```



Step 5: Displaying bar chart with Revenue

```
df$Revenue<-df$Price*df$Quantity
df <- df %>% arrange(Revenue)

df %>%
    group_by(Department) %>%
    summarise(Revenue = n()) %>%
    ggplot(aes(x = reorder(Department,(-Revenue)), y = Revenue)) +
  geom_bar(stat = 'identity', fill = "red", colour = "black") +
    labs(
      title = "Distribution of Revenues",
      y = "Revenue",
      x = "Department"
      ) +
    theme_bw()
```

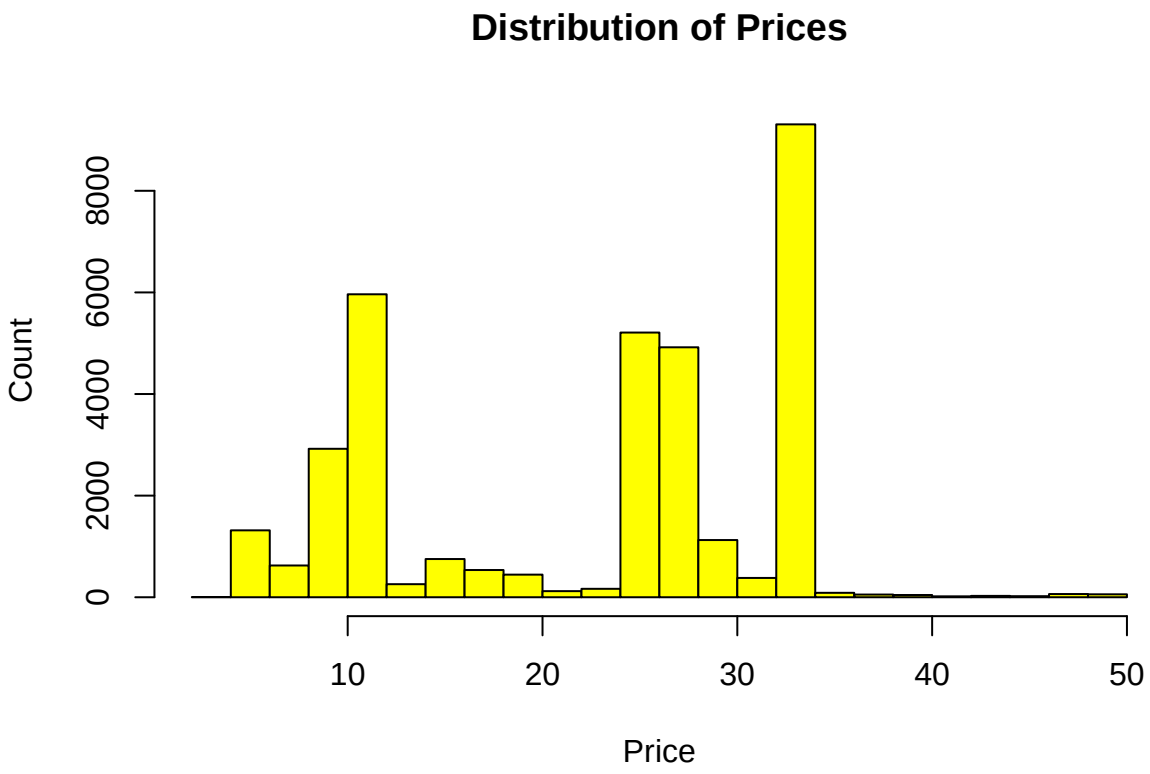


Step 6: Displaying Histograms. 6a: Boxplot of price

```
boxplot(df$Price, col = "yellow", main = "Distribution of Prices", ylab = "Count", xlab = "Price")
```

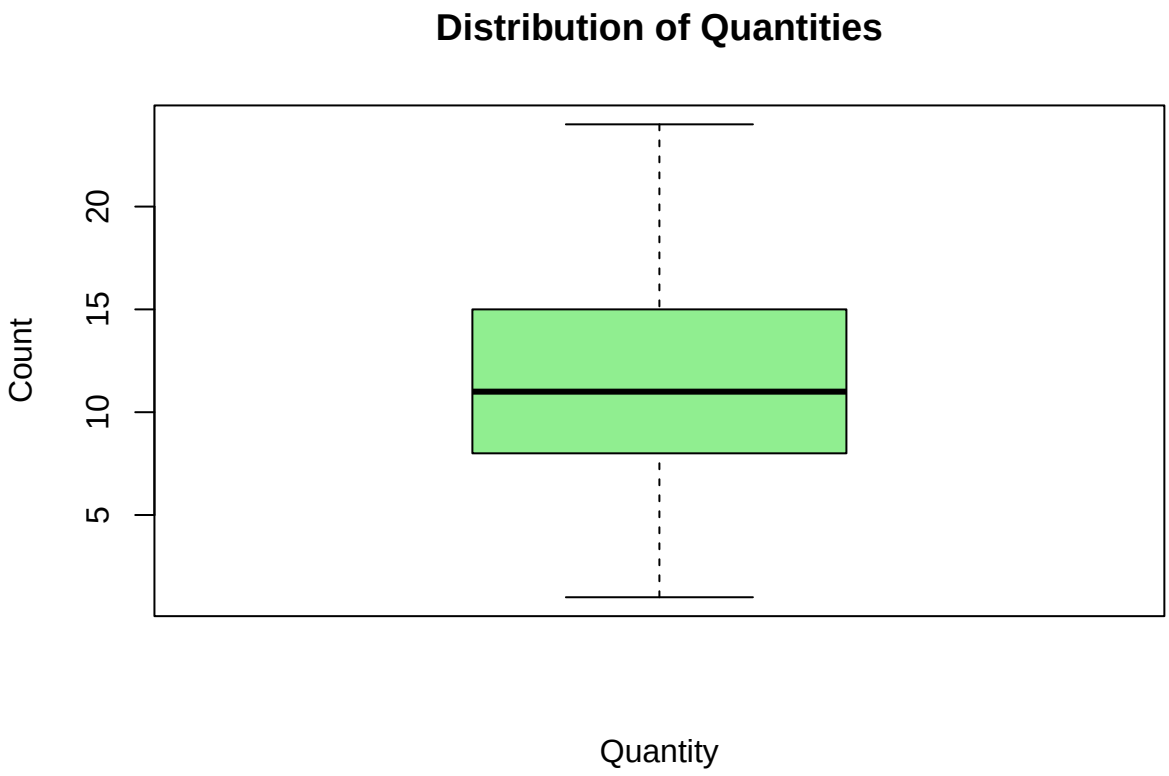
6b: histogram of price

```
hist(df$Price, col = "yellow", main = "Distribution of Prices", ylab = "Count", xlab = "Price")
```



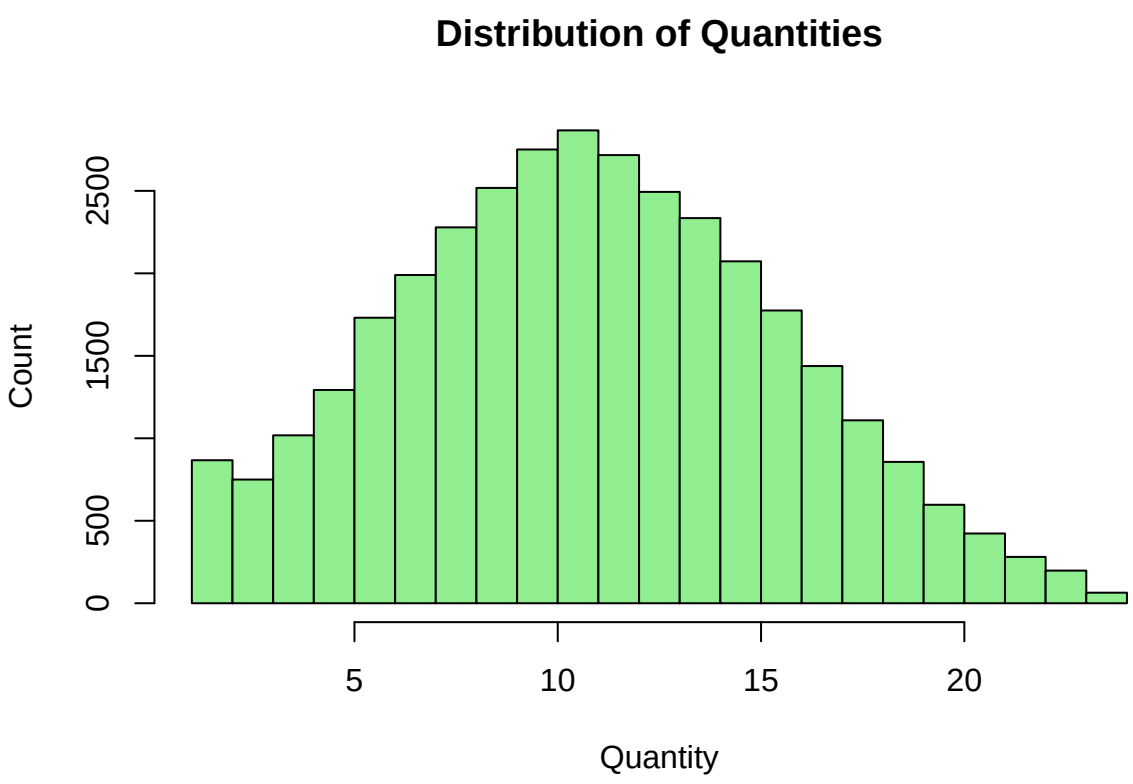
6c : Boxplot of Quantity

```
boxplot(df$Quantity, col = "lightgreen", main = "Distribution of Quantities", ylab = "Count", xlab = "Qu
```



6d : Histogram of Quantity

```
hist(df$Quantity, col = "lightgreen", main = "Distribution of Quantities", ylab = "Count", xlab = "Quant
```



Step 7: Small review

R is at best in terms of replicability, where it is easy to follow chunks of code, see them as used by others and re-adapt for your own interests. Steps can always be followed through easily as long as someone knows how to read them. Power BI and Arteryx do not follow the same case. This also means that R is very easy to share, which is the case for Power BI and Arteryx.

R is also completely free making it much better than Power BI in this sense, as the desktop version can be free with limited access only, and than Arteryx which provides you with a free trial only.

R is also better at dealing with large data sets, while Power BI and Arteryx handle better smaller data. Furthermore, Power BI and Arteryx are quite limited by being suitable only for Microsoft while R can easily be downloaded and executed in any operating system.