main

2024-06-10

Importing dataset and libraries

#EDA ##Data Cleaning

###Checking and Removing Duplicates

```
# Check for entire row duplicates
duplicates_entire_row <- duplicated(df)

# Count the number of entirely duplicated rows
num_entire_row_duplicates <- sum(duplicates_entire_row)
print(paste("Number of entirely duplicated rows:", num_entire_row_duplicates))
```

```
## [1] "Number of entirely duplicated rows: 0"
```

As we can see there aren't any duplicates in our data set, so we can proceed to find which columns have missing values

###Fixing Features of the Dataset

```
item_fat_content_uniques <- unique(df$Item_Fat_Content)#unique values of item_fat_content

print(item_fat_content_uniques)
```

```
## [1] "Low Fat" "Regular" "low fat" "LF"      "reg"
```

```
df$Item_Fat_Content <- ifelse(grepl("^[Rr]", df$Item_Fat_Content), "Regular Fat", "Low Fat")
item_fat_content_uniques <- unique(df$Item_Fat_Content)

print(item_fat_content_uniques)
```

```
## [1] "Low Fat"     "Regular Fat"
```

If we go throughout the column `Item_Fat_Content` we can cearly see that there are some inconsistencies, in fact we have 2 different names for "Regular Fat" that are "Regular" and "reg" and 3 different names for "Low Fat" that are "Low fat", "low fat", "LF"

To solve this problem, we edited the column "Item_Fat_Content" in this way: all records that start with R or r, will be renamed "Regular Fat", and other will be renamed "Low Fat".

Then we will print the uniques name of the column after the edit to chek if the process ended successfully

###Checking for Missing Values

```r
# Crea una copia del dataframe convertendo tutte le colonne in stringhe
df_copy <- lapply(df, as.character)

# Creiamo una funzione per contare "NA", "" o "0" in ogni colonna della copia
count_na_empty_or_zero_strings <- function(column) {
  # Sostituiamo i veri NA con stringhe vuote per una conta coerente
  column[is.na(column)] <- ""
  sum(column == "" | column == "0", na.rm = TRUE)
}

# Applicare la funzione a tutte le colonne della copia e stampare i risultati
na_empty_or_zero_counts <- sapply(df_copy, count_na_empty_or_zero_strings)

# Stampa il numero di "NA", "" o "0" per ogni colonna della copia
print(na_empty_or_zero_counts)
```

```
##           Item_Identifier                Item_Weight          Item_Fat_Content
##                         0                       1463                         0
##           Item_Visibility                  Item_Type                  Item_MRP
##                       526                          0                         0
##         Outlet_Identifier Outlet_Establishment_Year               Outlet_Size
##                         0                          0                      2410
##      Outlet_Location_Type                Outlet_Type         Item_Outlet_Sales
##                         0                          0                         0
```

We see that there are 3 columns with missing values: -Item_Weight -Item_Visibility -Outlet_Size

```r
#conteggio records senza Outlet_Size:
null<-sum(nchar(df$Outlet_Size) ==0)

#numero di records
total<- nrow(df)

print(paste("Number of rows with missing valuess on Outlet_Size:",null))
```

```
## [1] "Number of rows with missing valuess on Outlet_Size: 2410"
```

```r
#percentuale di righe senza outlet_size
percentuale_stringhe_vuote <- (null / total) * 100

#stampa percentuale di righe senza outlet_size

print(paste("Percentuale di records con stringhe vuote in 'Outlet_Size':", percentuale_stringhe_vuote,
```

```
## [1] "Percentuale di records con stringhe vuote in 'Outlet_Size': 28.2764284876217 %"
```

```r
#conteggio records senza Outlet_Size:
null<-sum(nchar(df$Outlet_Size) ==0)

#numero di records
total<- nrow(df)

print(paste("Number of rows with missing valuess on Outlet_Size:",null))
```

```
## [1] "Number of rows with missing valuess on Outlet_Size: 2410"
```

```
#percentuale di righe senza outlet_size
percentuale_stringhe_vuote <- (null / total) * 100

#stampa percentuale di righe senza outlet_size

print(paste("Percentuale di records con stringhe vuote in 'Outlet_Size':", percentuale_stringhe_vuote,
```

```
## [1] "Percentuale di records con stringhe vuote in 'Outlet_Size': 28.2764284876217 %"
```

```
# Aggiornamento della colonna 'Outlet_Size'
df <- df %>%
  mutate(Outlet_Size = case_when(
    Outlet_Type == "Grocery Store" ~ "Small",
    #Outlet_Type %in% c("Supermarket Type2", "Supermarket Type3") ~ "Medium",
    TRUE ~ Outlet_Size  # Mantiene il valore originale per tutte le altre condizioni
  ))

# Visualizza le modifiche per confermare
# Creazione di una tabella di riepilogo
outlet_summary <- df %>%
  filter(Outlet_Size %in% c("Small", "Medium", "High")) %>%  # Filtra per includere solo le righe con i
  group_by(Outlet_Type, Outlet_Size) %>%  # Raggruppa per tipo e dimensione del negozio
  summarise(Count = n(), .groups = 'drop')  # Calcola il conteggio e rimuove il raggruppamento automati

# Visualizzazione della tabella di riepilogo
print(outlet_summary)
```

```
## # A tibble: 6 x 3
##   Outlet_Type       Outlet_Size Count
##   <chr>             <chr>       <int>
## 1 Grocery Store     Small        1083
## 2 Supermarket Type1 High          932
## 3 Supermarket Type1 Medium        930
## 4 Supermarket Type1 Small        1860
## 5 Supermarket Type2 Medium        928
## 6 Supermarket Type3 Medium        935
```

```
# Aggiornamento della colonna 'Outlet_Size' per riempire le stringhe vuote
df <- df %>%
  mutate(Outlet_Size = if_else(nchar(Outlet_Size) == 0, "NA", Outlet_Size))
```

To address the significant number of missing values were noted in the `Outlet_Size` column, we first calculated the percentage of missing entries to understand the scope of the issue.

Subsequently, we explored potential relationships between `Outlet_Type` and `Outlet_Size` by creating a cross-tabulation of these variables. This analysis revealed distinct patterns: all entries categorized as "Grocery Store" were consistently labeled as "Small", while "Supermarket Type2" and "Supermarket Type3" were uniformly classified as "Medium".

Considering this insight, we proceeded to change missing `Outlet_Size` values based on `Outlet_Type`:

For “Grocery Store”, missing sizes were filled with “Small”. For “Supermarket Type2” and “Supermarket Type3”, missing sizes were filled with “Medium”. These imputations reduced the percentage of missing values from 28% to 21%, decreasing the total number of missing entries from 2410 to 1855.

Regarding “Supermarket Type1” we do not have enough informations or clear patterns, so we had to fill the remaining missing entries with a placeholder value of “NA”, ensuring no data point within “Outlet_Size” remained unaddressed.

```
# Conversione di 'Outlet_Size' in valori numerici
df <- df %>%
  mutate(Outlet_Size = case_when(
    Outlet_Size == "Small" ~ 1,
    Outlet_Size == "Medium" ~ 2,
    Outlet_Size == "High" ~ 3,
    TRUE ~ NA_real_  # Imposta NA per qualsiasi altro valore non specificato
  ))

df <- df%>%
  mutate(Item_Fat_Content = case_when(
    Item_Fat_Content == "Low Fat" ~ 1,
    Item_Fat_Content == "Regular Fat" ~ 2,
    TRUE ~ NA_real_
  ))
head(df)
```

```
##   Item_Identifier Item_Weight Item_Fat_Content Item_Visibility
## 1           FDA15       9.300                1      0.01604730
## 2           DRC01       5.920                2      0.01927822
## 3           FDN15      17.500                1      0.01676007
## 4           FDX07      19.200                2      0.00000000
## 5           NCD19       8.930                1      0.00000000
## 6           FDP36      10.395                2      0.00000000
##               Item_Type Item_MRP Outlet_Identifier Outlet_Establishment_Year
## 1                 Dairy 249.8092            OUT049                      1999
## 2           Soft Drinks  48.2692            OUT018                      2009
## 3                  Meat 141.6180            OUT049                      1999
## 4 Fruits and Vegetables 182.0950            OUT010                      1998
## 5             Household  53.8614            OUT013                      1987
## 6          Baking Goods  51.4008            OUT018                      2009
##   Outlet_Size Outlet_Location_Type       Outlet_Type Item_Outlet_Sales
## 1           2               Tier 1 Supermarket Type1         3735.1380
## 2           2               Tier 3 Supermarket Type2          443.4228
## 3           2               Tier 1 Supermarket Type1         2097.2700
## 4           1               Tier 3     Grocery Store          732.3800
## 5           3               Tier 3 Supermarket Type1          994.7052
## 6           2               Tier 3 Supermarket Type2          556.6088
```

We converted as factor the columns `Item_Fat_Content` and we also want to convert the column `Outlet_Size` giving respectively: 0 -> “Low Fat” 1 -> “Regular”

1 -> Small 2 -> Medium 3 -> Large

```
# Calcolare la media del peso per ogni categoria di prodotto
average_weight_per_type <- aggregate(Item_Weight ~ Item_Type, data = df, mean, na.rm = TRUE)
```

```r
# Funzione per riempire i pesi mancanti
fill_missing_weights <- function(item_id, item_type) {
  # Controlla se esiste un valore non NA per lo stesso Item_Identifier
  if (any(!is.na(df$Item_Weight[df$Item_Identifier == item_id]))) {
    return(df$Item_Weight[df$Item_Identifier == item_id & !is.na(df$Item_Weight)][1])
  } else {
    # Altrimenti usa la media del Item_Type corrispondente
    return(average_weight_per_type$Item_Weight[average_weight_per_type$Item_Type == item_type])
  }
}

# Applicare la funzione ai valori NA in Item_Weight
df$Item_Weight[is.na(df$Item_Weight)] <- mapply(fill_missing_weights, df$Item_Identifier[is.na(df$Item_

# Visualizzare il dataframe aggiornato
head(df)
```

```
##   Item_Identifier Item_Weight Item_Fat_Content Item_Visibility
## 1           FDA15       9.300               1      0.01604730
## 2           DRC01       5.920               2      0.01927822
## 3           FDN15      17.500               1      0.01676007
## 4           FDX07      19.200               2      0.00000000
## 5           NCD19       8.930               1      0.00000000
## 6           FDP36      10.395               2      0.00000000
##               Item_Type Item_MRP Outlet_Identifier Outlet_Establishment_Year
## 1                 Dairy 249.8092            OUT049                      1999
## 2           Soft Drinks  48.2692            OUT018                      2009
## 3                  Meat 141.6180            OUT049                      1999
## 4 Fruits and Vegetables 182.0950            OUT010                      1998
## 5             Household  53.8614            OUT013                      1987
## 6          Baking Goods  51.4008            OUT018                      2009
##   Outlet_Size Outlet_Location_Type       Outlet_Type Item_Outlet_Sales
## 1           2               Tier 1 Supermarket Type1         3735.1380
## 2           2               Tier 3 Supermarket Type2          443.4228
## 3           2               Tier 1 Supermarket Type1         2097.2700
## 4           1               Tier 3     Grocery Store          732.3800
## 5           3               Tier 3 Supermarket Type1          994.7052
## 6           2               Tier 3 Supermarket Type2          556.6088
```

Here we handled missing values for `Item_Weight`. The first idea was to use the general mean of `Item_Weight`, but since we have also the category of each product, we can compute the mean of every product category. At the end, we checked for every record with missing `Item_Weight` and we will do a double check: 1. If there were another item with the same id and missing weight, they will have for sure the same weight. 2. We will put in item_weight, the mean of the weights of the products of the same category.

```r
# Find the number of zero 'Item_Visibility' values for each 'Outlet_Identifier'
zero_visibility_counts <- aggregate(Item_Visibility ~ Outlet_Identifier, data = df, function(x) sum(x ==

# Rename the column for better understanding
names(zero_visibility_counts)[2] <- "Zero_Item_Visibility_Count"

# Display the result
print(zero_visibility_counts)
```

```
##    Outlet_Identifier Zero_Item_Visibility_Count
## 1             OUT010                        29
## 2             OUT013                        59
## 3             OUT017                        57
## 4             OUT018                        65
## 5             OUT019                        30
## 6             OUT027                        60
## 7             OUT035                        54
## 8             OUT045                        58
## 9             OUT046                        61
## 10            OUT049                        53
```

```r
visibility_sum_per_store <- aggregate(Item_Visibility ~ Outlet_Identifier, data = df, sum)
names(visibility_sum_per_store)[2] <- "Total_Item_Visibility"

# Display the result
print(visibility_sum_per_store)
```

```
##    Outlet_Identifier Total_Item_Visibility
## 1             OUT010              56.30883
## 2             OUT013              55.87986
## 3             OUT017              56.83465
## 4             OUT018              56.62145
## 5             OUT019              57.25704
## 6             OUT027              54.80476
## 7             OUT035              56.97487
## 8             OUT045              56.18078
## 9             OUT046              56.23188
## 10            OUT049              56.54916
```

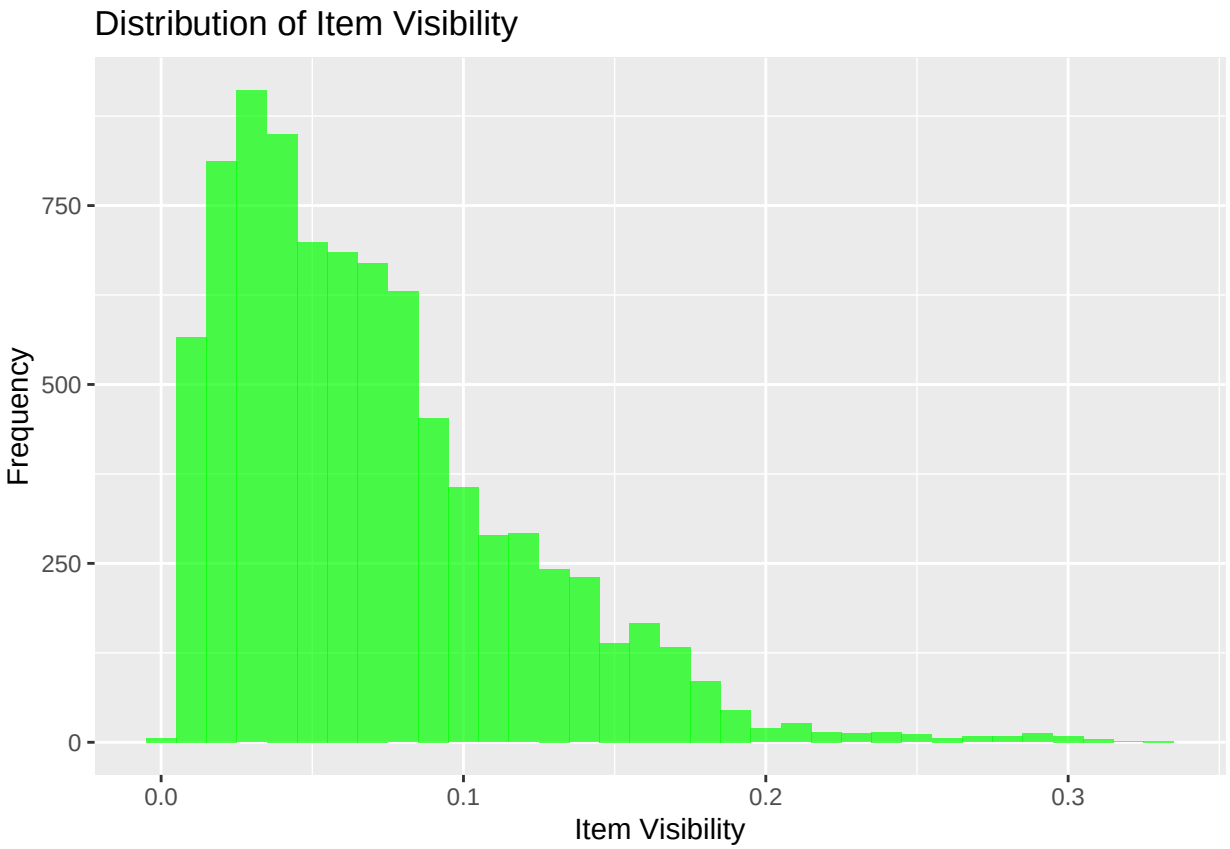
```r
# Caricare il dataset

# Filtrare i record con visibilità maggiore di zero per calcolare le medie
filtered_df <- df[df$Item_Visibility > 0, ]

# Calcolare la media di Item_Visibility per ogni combinazione di Item_Type e Outlet_Size
visibility_avg_per_type_size <- aggregate(Item_Visibility ~ Item_Type + Outlet_Size, data = filtered_df

# Creare una chiave unica per facilitare il merge
visibility_avg_per_type_size$key <- with(visibility_avg_per_type_size, paste(Item_Type, Outlet_Size, se
df$key <- with(df, paste(Item_Type, Outlet_Size, sep = "_"))

# Merge tra i record del dataframe originale e le medie calcolate usando un left join
df <- merge(df, visibility_avg_per_type_size, by = "key", all.x = TRUE, suffixes = c("", ".new"))

# Sostituire i valori zero di Item_Visibility con i valori medi calcolati
df$Item_Visibility[df$Item_Visibility == 0] <- df$Item_Visibility.new[df$Item_Visibility == 0]

# Rimuovere le colonne in più create dal merge
df <- df[, !names(df) %in% c("Item_Visibility.new", "key")]

# Visualizzare il dataframe aggiornato
head(df)
```

```
##   Item_Identifier Item_Weight Item_Fat_Content Item_Visibility    Item_Type
## 1           FDA11       7.750                1     0.043238822 Baking Goods
## 2           FDA36       5.985                1     0.009921107 Baking Goods
## 3           FDQ12      12.650                1     0.035404052 Baking Goods
## 4           FDO12      15.750                1     0.054920146 Baking Goods
## 5           FDM60      10.800                2     0.048143292 Baking Goods
## 6           FDC48       9.195                1     0.015856295 Baking Goods
##   Item_MRP Outlet_Identifier Outlet_Establishment_Year Outlet_Size
## 1  92.5436            OUT046                      1997           1
## 2 183.6924            OUT019                      1985           1
## 3 230.6010            OUT035                      2004           1
## 4 195.8452            OUT035                      2004           1
## 5  40.2138            OUT046                      1997           1
## 6  81.6592            OUT035                      2004           1
##   Outlet_Location_Type       Outlet_Type Item_Outlet_Sales Item_Type.new
## 1               Tier 1 Supermarket Type1         1701.7848  Baking Goods
## 2               Tier 1     Grocery Store          555.2772  Baking Goods
## 3               Tier 2 Supermarket Type1         2067.3090  Baking Goods
## 4               Tier 2 Supermarket Type1         4893.6300  Baking Goods
## 5               Tier 1 Supermarket Type1          690.4346  Baking Goods
## 6               Tier 2 Supermarket Type1         1403.5064  Baking Goods
##   Outlet_Size.new
## 1               1
## 2               1
## 3               1
## 4               1
## 5               1
## 6               1
```

```
df <- df %>%
  select(-Item_Type.new, -Outlet_Size.new)
```

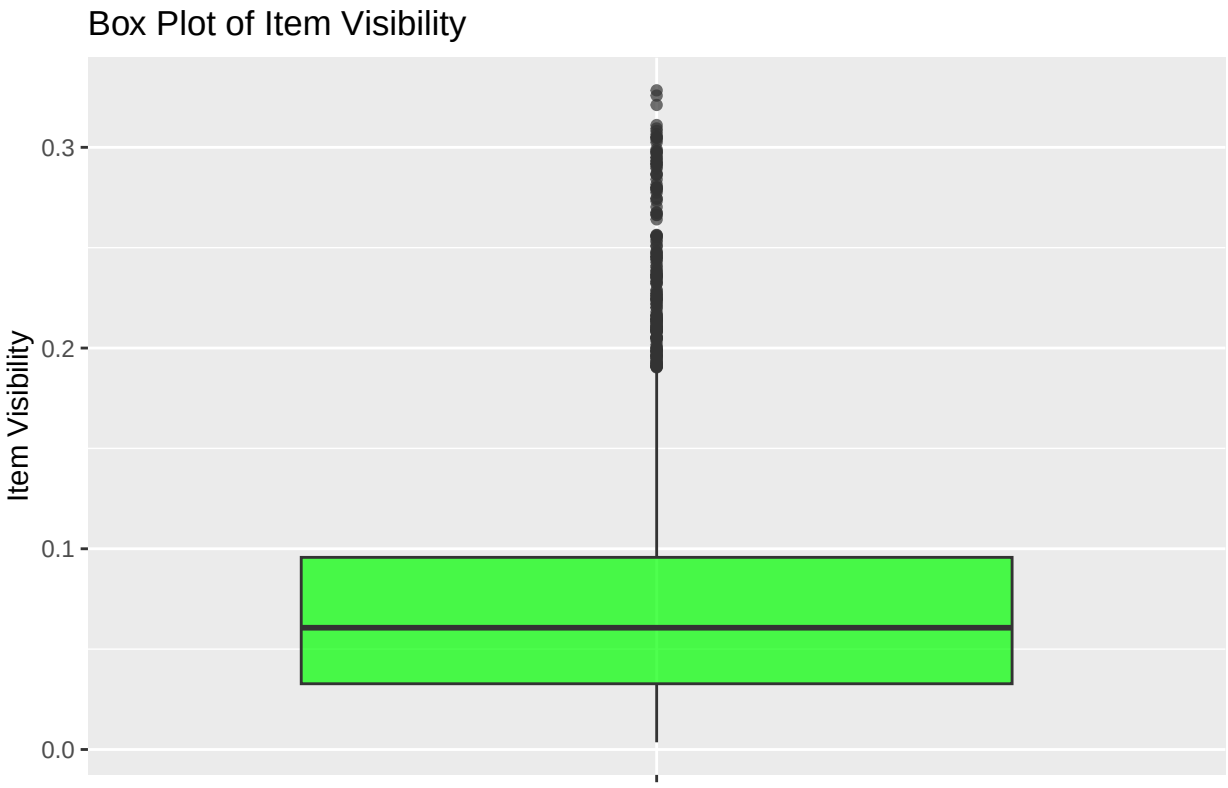
```
# Histogram and Box Plot for Item_Visibility
p3 <- ggplot(df, aes(x = Item_Visibility)) +
  geom_histogram(binwidth = 0.01, fill = 'green', alpha = 0.7) +
  ggtitle("Distribution of Item Visibility") +
  xlab("Item Visibility") +
  ylab("Frequency")

p4 <- ggplot(df, aes(x = "", y = Item_Visibility)) +
  geom_boxplot(fill = 'green', alpha = 0.7) +
  ggtitle("Box Plot of Item Visibility") +
  xlab("") +
  ylab("Item Visibility")
print(p3)
```

```
## Warning: Removed 115 rows containing non-finite outside the scale range
## (`stat_bin()`).
```




```
print(p4)
```

```
## Warning: Removed 115 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```



Finally we fixed the missing values for `Item_Visibility`. To do it we firstly count the number of missing values for each of the 10 shops and their visibility to understand how much visibility we are missing from each store. Our first idea was to subtract each visibility percentage from 100% and then equally divide the remaining visibility among the number of missing visibility product. In doing that we obtained biased and not consistent data because:

1. The filled visibility in each product was too high compared to the non-zeros values
2. After adding again all the visibility the result was 100% for each shop, but could also be true that in some shops some products are out of stock so the final visibility could be less than 100%

For this reason we decided to change our strategy and to fill the missing values we firstly find the missing `Item_Visibility`, then through the `Item_ID` we find all the shops that have for that product a real value, then we check the `Outlet_Size` of each shop and compare to the one of the market with missing value if it's the same we compute the mean throu all the visibility that match these criteria.

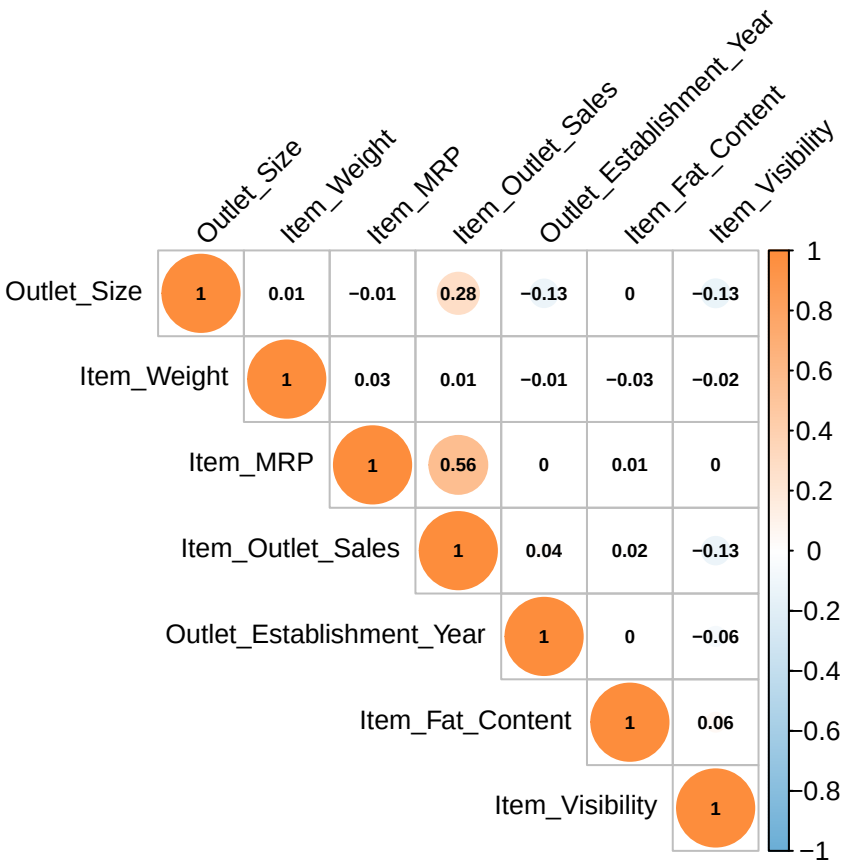
We did this because for example if the missing visibility is in a Medium shop, we cannot consider the same product visibility in a Small shop, because it's obviously higher, so we take into accounts a visibility to compute the mean only if the `Outlet_Size` is the same.

##Multivariate Analysis

Now we will start to find out some correlations between variables.

```
# Calculate Spearman's rank correlation matrix again if not already calculated
cor_matrix <- cor(df %>% select(where(is.numeric)),
                  method = "spearman",
                  use = "pairwise.complete.obs")
```

```r
# Visualize the correlation matrix with coefficients inside the circles
corrplot(cor_matrix, method = "circle", type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45, tl.cex = 0.8, cl.cex = 0.8,
         col = colorRampPalette(c("#6BAED6", "#FFFFFF", "#FD8D3C"))(200),
         addCoef.col = "black",  # Sets color of the coefficients to black (choose based on your color
         number.cex = 0.6)  # Adjust coefficient text size appropriately
```

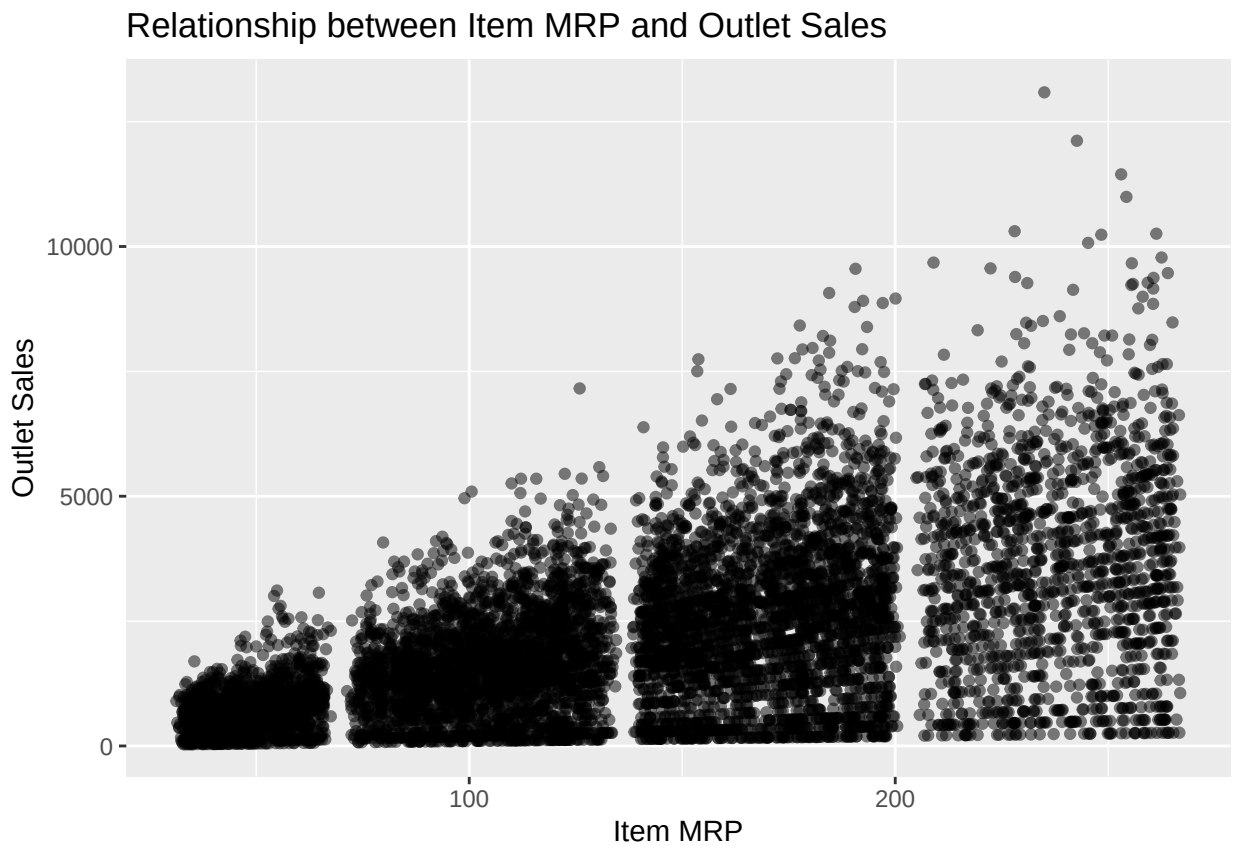


DOBBIAMO DROPPARE LE COLONNE REMINAINING VISIBILITY EZERO ITEM VISIBILITY COUNT

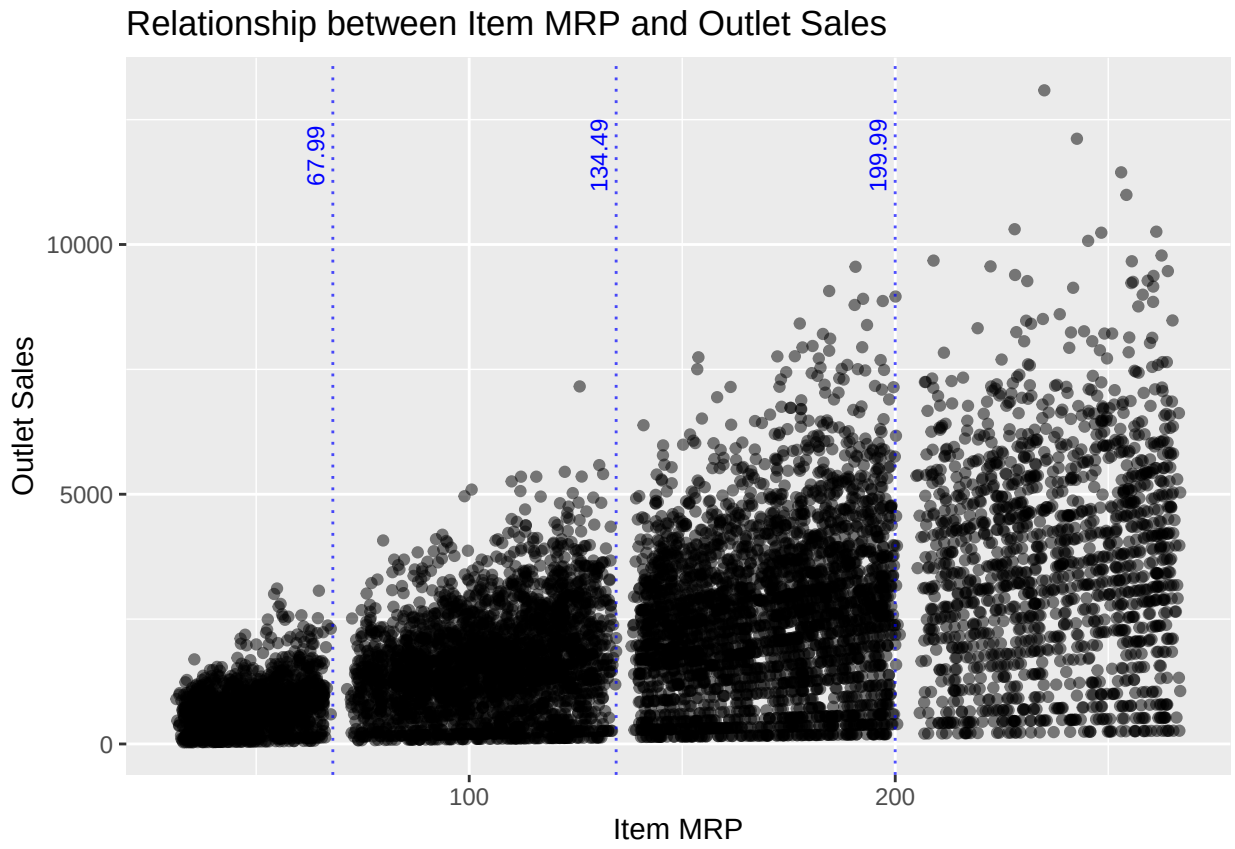
We can see that there are big positive relation between:

outlet_size-item visibility item_outlet_sales and outlet_size item_mrp and item_outlet_sales

```r
library(ggplot2)
ggplot(df, aes(x = Item_MRP, y = Item_Outlet_Sales)) +
  geom_point(alpha = 0.5) +
  labs(title = "Relationship between Item MRP and Outlet Sales",
       x = "Item MRP", y = "Outlet Sales")
```




```r
# Load necessary libraries
library(ggplot2)

# Specific price points to add to the plot
specific_price_points <- c(67.99, 134.49, 199.99)

# Plot the relationship between Item_MRP and Item_Outlet_Sales
p <- ggplot(df, aes(x = Item_MRP, y = Item_Outlet_Sales)) +
  geom_point(alpha = 0.5) +
  labs(title = "Relationship between Item MRP and Outlet Sales",
       x = "Item MRP", y = "Outlet Sales")

# Add vertical lines for specific price points
for (price in specific_price_points) {
  p <- p + geom_vline(xintercept = price, linetype = "dotted", color = "blue", alpha = 0.7)
  p <- p + annotate("text", x = price, y = max(df$Item_Outlet_Sales) * 0.9, label = price, color = "blu
}

# Print the plot
print(p)
```

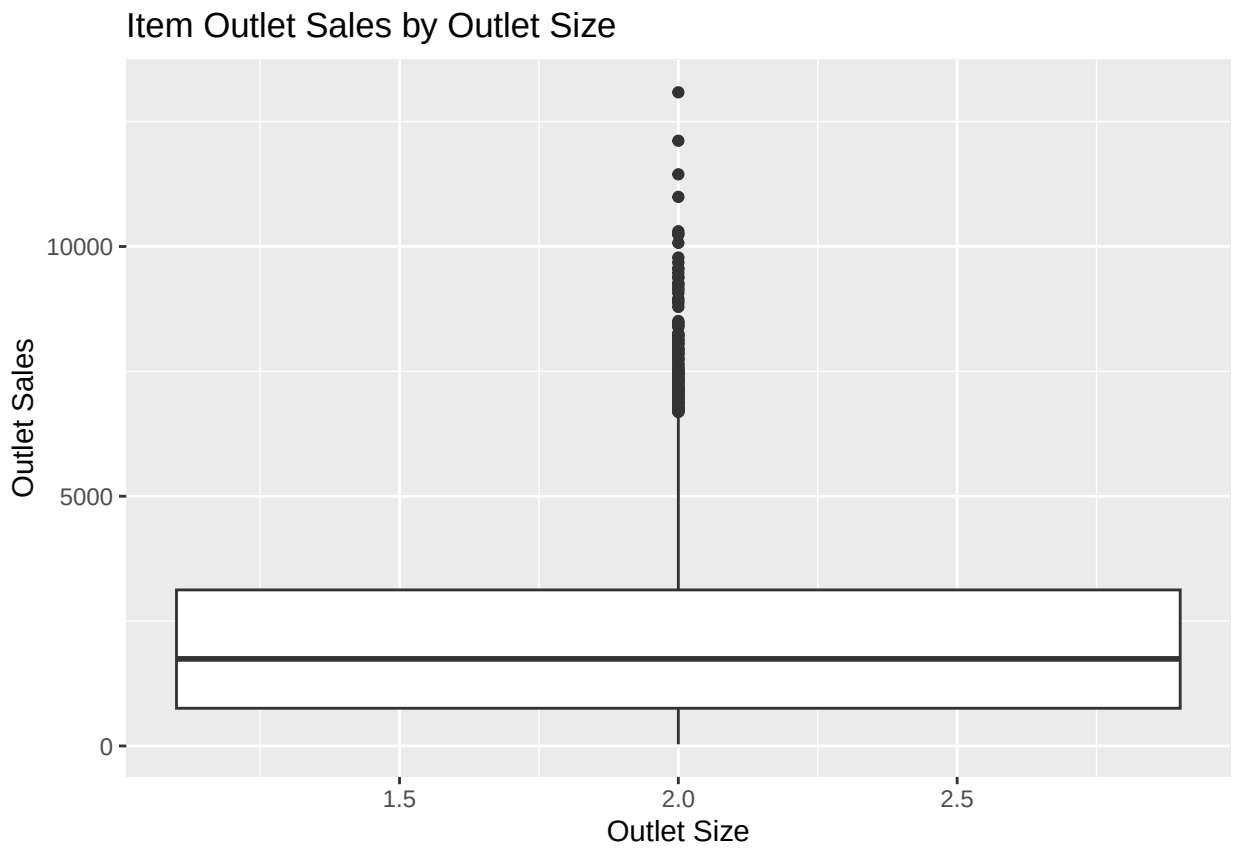



```
ggplot(df, aes(x = Outlet_Size, y = Item_Outlet_Sales)) +
  geom_boxplot(aes(fill = Outlet_Size)) +
  labs(title = "Item Outlet Sales by Outlet Size",
       x = "Outlet Size", y = "Outlet Sales")
```

```
## Warning: Continuous x aesthetic
## i did you forget `aes(group = ...)`?
```

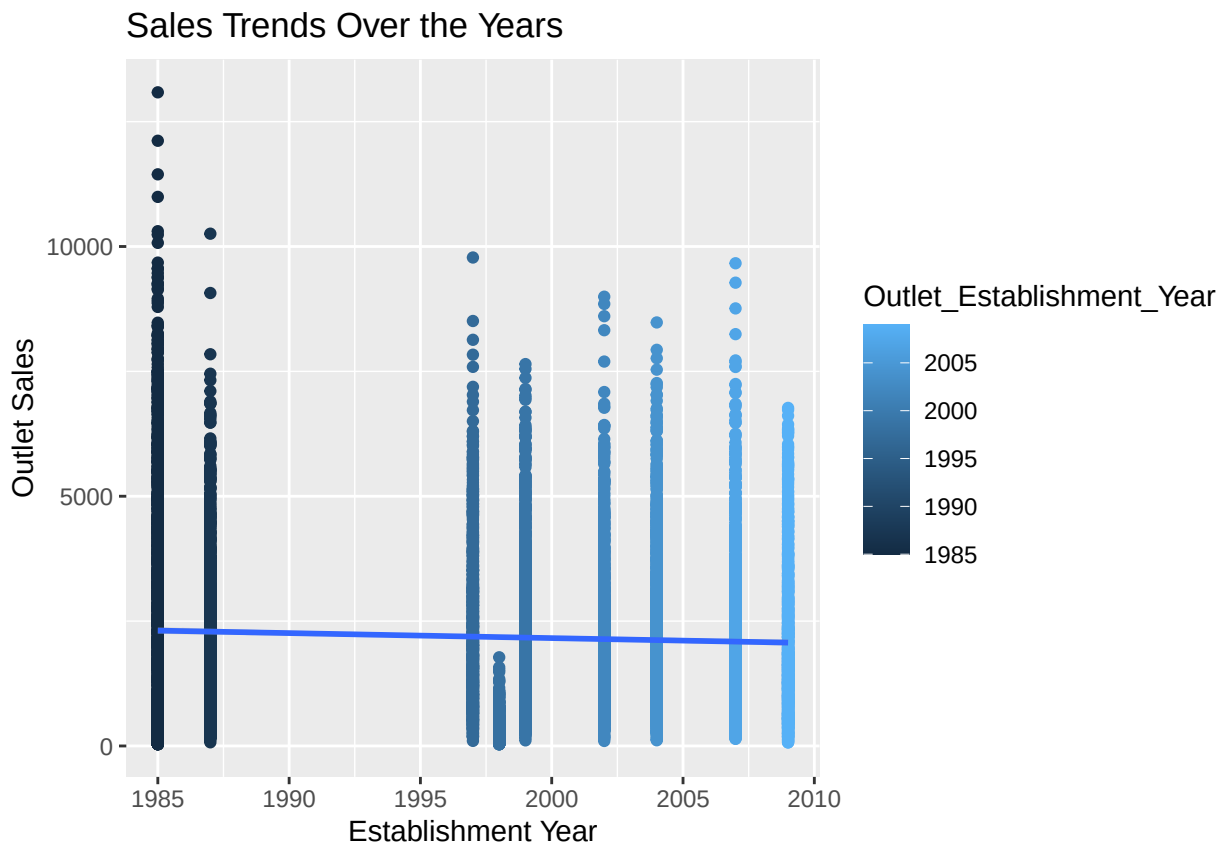
```
## Warning: Removed 1855 rows containing missing values or values outside the scale range
## (`stat_boxplot()`).
```

```
## Warning: The following aesthetics were dropped during statistical transformation: fill.
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
```

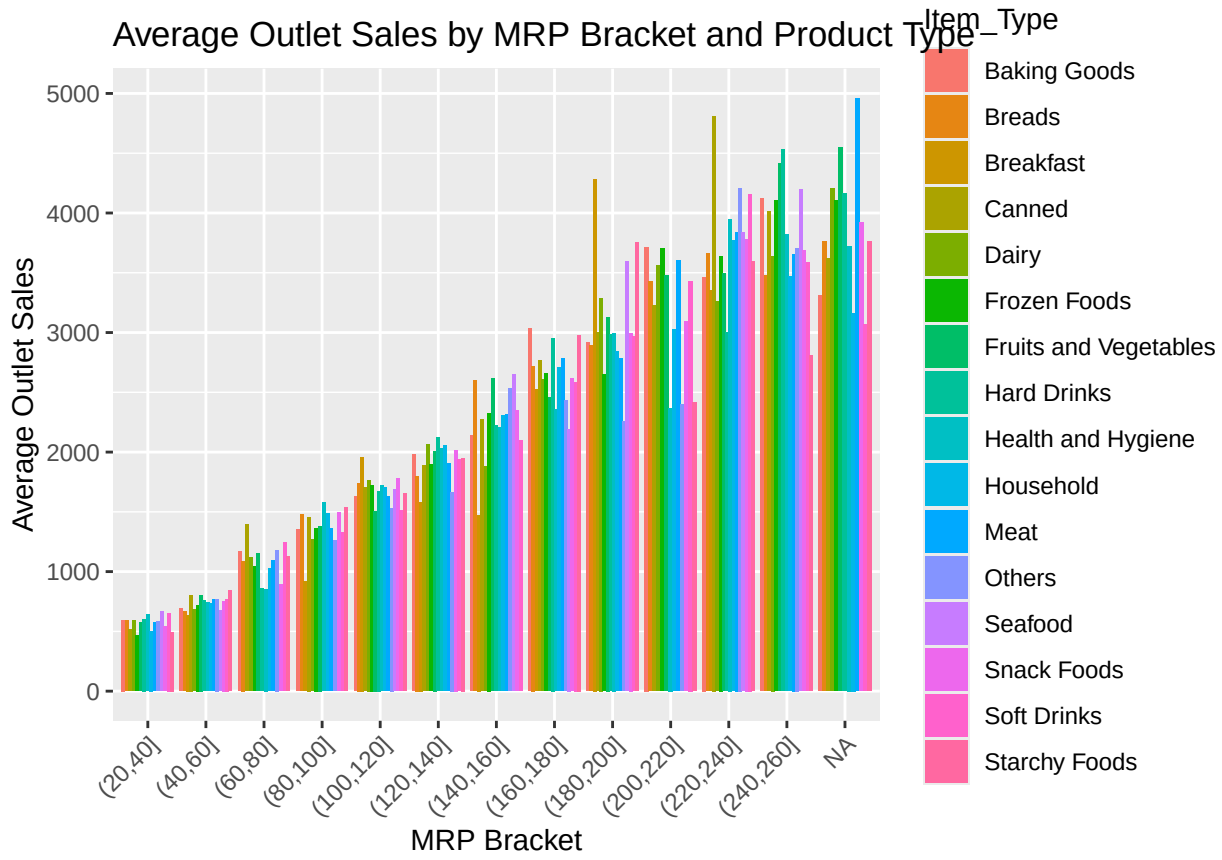



```
ggplot(df, aes(x = Outlet_Establishment_Year, y = Item_Outlet_Sales)) +
  geom_point(aes(color = Outlet_Establishment_Year)) +
  geom_smooth(method = "lm") +
  labs(title = "Sales Trends Over the Years",
       x = "Establishment Year", y = "Outlet Sales")
```

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



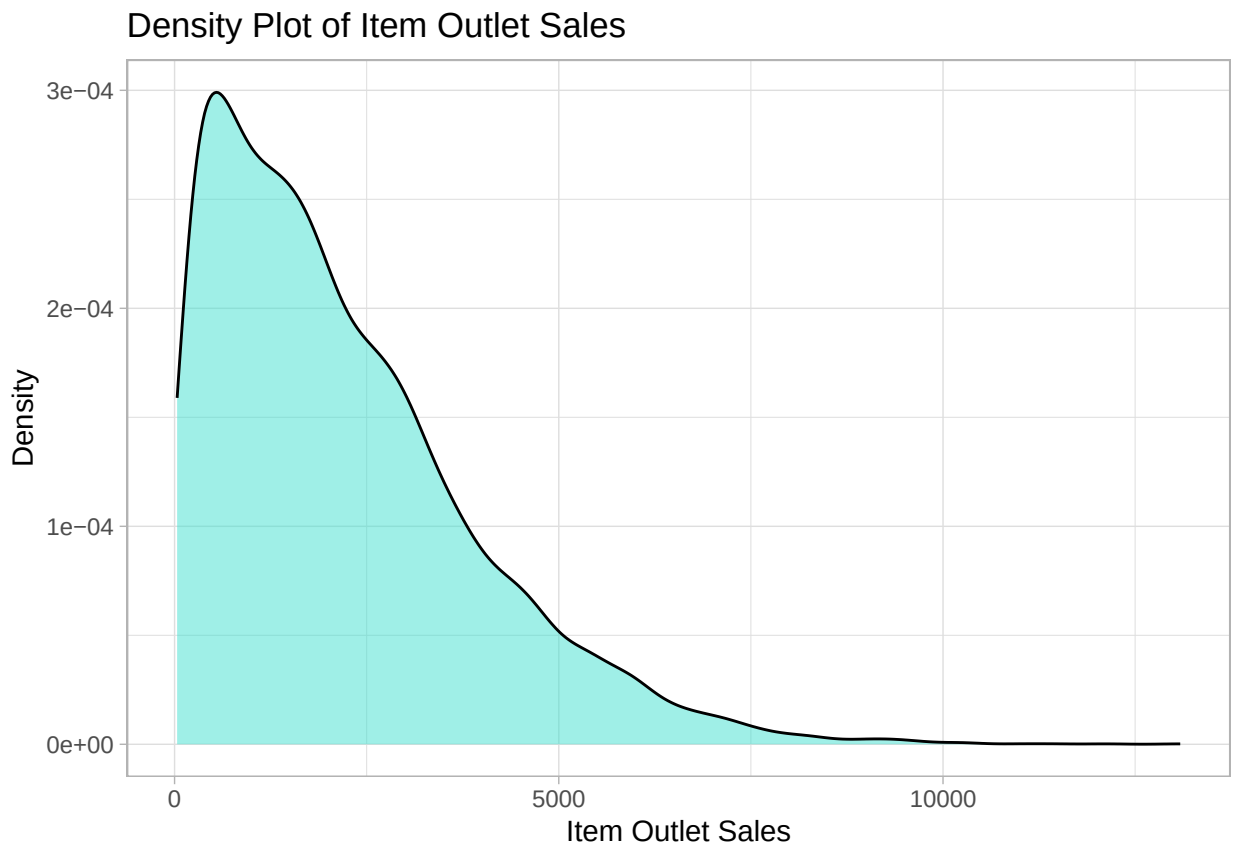

```
library(dplyr)

df_summary <- df %>%
  group_by(Item_Type, MRP_Bracket = cut(Item_MRP, breaks = seq(0, max(Item_MRP), by = 20))) %>%
  summarize(Average_Sales = mean(Item_Outlet_Sales), .groups = 'drop')

ggplot(data = df_summary, aes(x = MRP_Bracket, y = Average_Sales, fill = Item_Type)) +
  geom_bar(stat = "identity", position = position_dodge()) +
  labs(title = "Average Outlet Sales by MRP Bracket and Product Type",
       x = "MRP Bracket",
       y = "Average Outlet Sales") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
#ciaociao
#namoooo
#prova 3
```

zi qua c'è scritto che più vendi e più alzi il prezzo di base

```
# Create a density plot
ggplot(data = df, aes(x = Item_Outlet_Sales)) +
  geom_density(fill = "turquoise", alpha = 0.5) +  # 'alpha' controls transparency
  labs(title = "Density Plot of Item Outlet Sales",
       x = "Item Outlet Sales",
       y = "Density") +
  theme_light()
```




```
#ciao2
```

```
# Ensure 'Item_Visibility' and 'Outlet_Sales' are treated correctly
df$Item_Visibility <- as.numeric(as.character(df$Item_Visibility))
df$Item_Outlet_Sales <- as.numeric(as.character(df$Item_Outlet_Sales))

# Convert categorical variables to factors
df$Item_Type <- as.factor(df$Item_Type)
df$Outlet_Type <- as.factor(df$Outlet_Type)

# Segmented Regression by Item Type
item_type_models <- df %>%
  group_by(Item_Type) %>%
  do(model = lm(Item_Outlet_Sales ~ Item_Visibility, data = .))

# Viewing summaries for each Item Type
item_type_summaries <- lapply(item_type_models$model, summary)

# Print the summaries for review
print(item_type_summaries)
```

```
## [[1]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2164.3 -1118.6  -420.1   849.9  5701.1
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2413.1      108.9   22.17  < 2e-16 ***
## Item_Visibility  -6210.5     1222.6   -5.08 4.97e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1517 on 639 degrees of freedom
##   (7 observations deleted due to missingness)
## Multiple R-squared:  0.03881,    Adjusted R-squared:  0.03731
## F-statistic:  25.8 on 1 and 639 DF,  p-value: 4.97e-07
##
##
## [[2]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2519.1 -1286.8  -269.4   881.4  6364.4
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2752.6      190.7  14.436  < 2e-16 ***
## Item_Visibility  -7701.9     2256.6  -3.413 0.000752 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1622 on 244 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.04557,    Adjusted R-squared:  0.04166
## F-statistic: 11.65 on 1 and 244 DF,  p-value: 0.0007519
##
##
## [[3]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2327.2 -1419.0  -510.4   815.1  5902.5
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2887.7      357.2   8.085  9.8e-13 ***
## Item_Visibility  -8810.8     3516.5  -2.506   0.0137 *
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1867 on 108 degrees of freedom
## Multiple R-squared:  0.05493,    Adjusted R-squared:  0.04618
## F-statistic: 6.278 on 1 and 108 DF,  p-value: 0.01372
##
##
## [[4]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2264.3 -1242.3  -402.4   873.7  7983.5
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2411.0      109.6  21.991   <2e-16 ***
## Item_Visibility  -2864.0     1240.8  -2.308   0.0213 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1623 on 641 degrees of freedom
##   (6 observations deleted due to missingness)
## Multiple R-squared:  0.008243,   Adjusted R-squared:  0.006696
## F-statistic: 5.328 on 1 and 641 DF,  p-value: 0.02131
##
##
## [[5]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2515.0 -1345.5  -530.1  1021.8  7711.3
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)         2767        130  21.279  < 2e-16 ***
## Item_Visibility    -6974       1415  -4.928 1.05e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1856 on 670 degrees of freedom
##   (10 observations deleted due to missingness)
## Multiple R-squared:  0.03498,    Adjusted R-squared:  0.03354
## F-statistic: 24.29 on 1 and 670 DF,  p-value: 1.048e-06
##
##
## [[6]]
##
## Call:
```

```
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2405.0 -1333.7  -436.5   943.4  7209.7
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2562.0      106.3  24.098  < 2e-16 ***
## Item_Visibility  -6084.6     1268.4  -4.797  1.9e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1703 on 842 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.0266, Adjusted R-squared:  0.02545
## F-statistic: 23.01 on 1 and 842 DF,  p-value: 1.902e-06
##
##
## [[7]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2512.9 -1341.4  -438.9   945.5  9504.5
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2691.51      91.64   29.37  < 2e-16 ***
## Item_Visibility -5488.80    1031.76   -5.32 1.24e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1782 on 1215 degrees of freedom
##   (15 observations deleted due to missingness)
## Multiple R-squared:  0.02276,    Adjusted R-squared:  0.02196
## F-statistic:  28.3 on 1 and 1215 DF,  p-value: 1.236e-07
##
##
## [[8]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2159.6 -1305.9  -357.7   938.3  5715.1
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2326.2      202.8  11.469   <2e-16 ***
## Item_Visibility  -2755.7     2397.9  -1.149    0.252
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1618 on 207 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.00634,    Adjusted R-squared:  0.00154
## F-statistic: 1.321 on 1 and 207 DF,  p-value: 0.2518
##
##
## [[9]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2135.5 -1201.3  -325.7   814.9  7773.3
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2351.5      116.8  20.136  < 2e-16 ***
## Item_Visibility  -5763.4     1612.1  -3.575 0.000383 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1543 on 513 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.02431,    Adjusted R-squared:  0.02241
## F-statistic: 12.78 on 1 and 513 DF,  p-value: 0.0003831
##
##
## [[10]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2397.0 -1323.1  -284.6   976.8 10548.5
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2592.96      91.55  28.323  < 2e-16 ***
## Item_Visibility -5169.13    1113.77  -4.641 3.98e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1677 on 893 degrees of freedom
##   (15 observations deleted due to missingness)
## Multiple R-squared:  0.02355,    Adjusted R-squared:  0.02246
## F-statistic: 21.54 on 1 and 893 DF,  p-value: 3.984e-06
##
##
## [[11]]
```

```
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2185.5 -1365.2  -368.9   861.1  7172.5
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2288.6      133.9   17.09   <2e-16 ***
## Item_Visibility  -1925.5     1604.5   -1.20    0.231
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1699 on 416 degrees of freedom
##   (7 observations deleted due to missingness)
## Multiple R-squared:  0.00345,    Adjusted R-squared:  0.001054
## F-statistic:  1.44 on 1 and 416 DF,  p-value: 0.2308
##
##
## [[12]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1886.8 -1135.1  -219.4   798.5  4082.7
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1933.7      186.7  10.357   <2e-16 ***
## Item_Visibility   -127.1     2334.9  -0.054    0.957
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1440 on 166 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  1.784e-05,  Adjusted R-squared:  -0.006006
## F-statistic: 0.002961 on 1 and 166 DF,  p-value: 0.9567
##
##
## [[13]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2252.9 -1467.9  -299.1  1242.9  4136.6
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)       2583.4      396.7   6.512 1.69e-08 ***
## Item_Visibility  -4006.8     4028.0  -0.995    0.324
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1812 on 60 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.01622,    Adjusted R-squared:  -0.0001724
## F-statistic: 0.9895 on 1 and 60 DF,  p-value: 0.3239
##
##
## [[14]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2455.4 -1295.6  -380.7   824.9  8800.4
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2591.36      87.24  29.705  < 2e-16 ***
## Item_Visibility -4481.25    1014.99  -4.415  1.1e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1695 on 1186 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.01617,    Adjusted R-squared:  0.01534
## F-statistic: 19.49 on 1 and 1186 DF,  p-value: 1.102e-05
##
##
## [[15]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2046.6 -1222.8  -529.1   725.7  7453.7
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2149.1      140.2  15.328   <2e-16 ***
## Item_Visibility  -2068.1     1653.8  -1.251    0.212
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1680 on 433 degrees of freedom
##   (10 observations deleted due to missingness)
## Multiple R-squared:  0.003598,   Adjusted R-squared:  0.001297
## F-statistic: 1.564 on 1 and 433 DF,  p-value: 0.2118
##
```

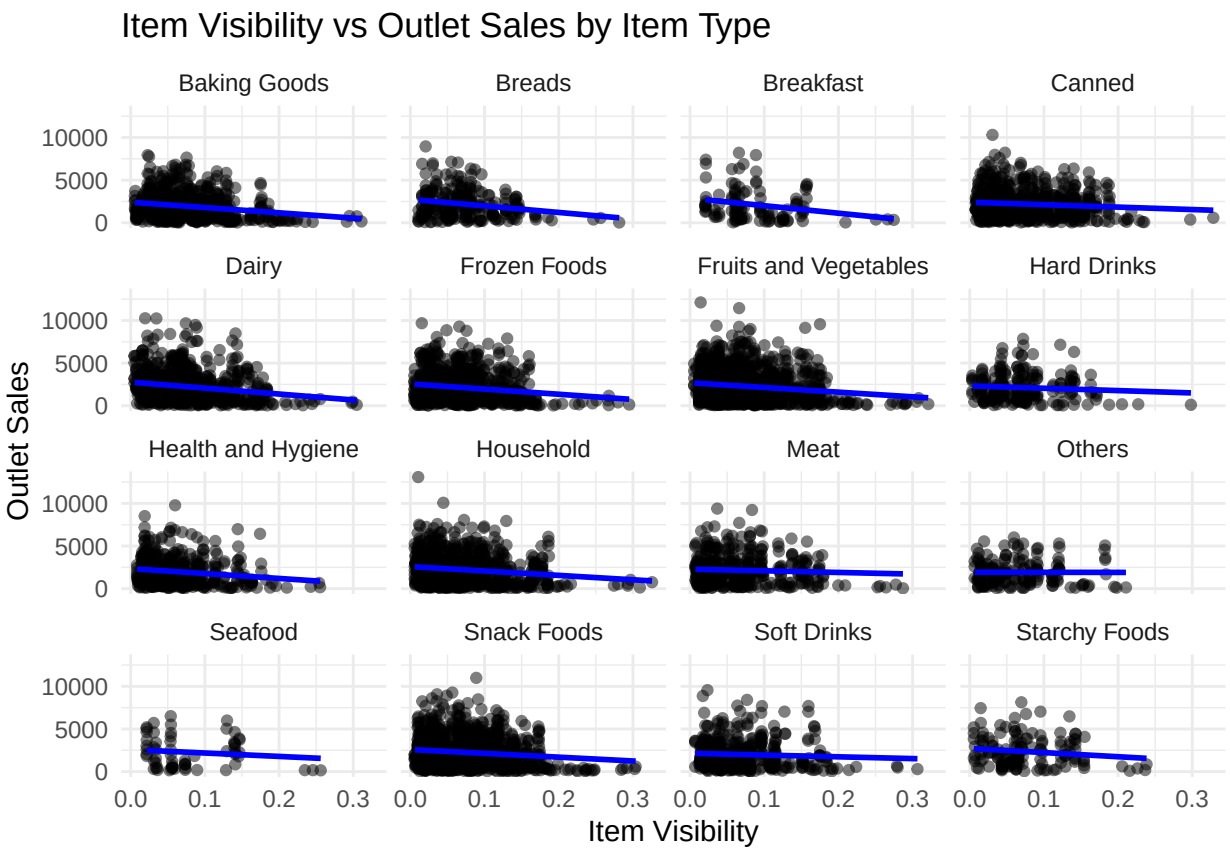
```
##
## [[16]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2353.4 -1393.4  -424.3  1145.1  5751.7
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2719.0      263.2  10.330   <2e-16 ***
## Item_Visibility  -4863.7     2906.8  -1.673   0.0965 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1742 on 143 degrees of freedom
##   (3 observations deleted due to missingness)
## Multiple R-squared:  0.0192, Adjusted R-squared:  0.01234
## F-statistic:   2.8 on 1 and 143 DF,  p-value: 0.09647
```

```
# Creating scatter plots segmented by Item_Type
p_item_type <- ggplot(df, aes(x = Item_Visibility, y = Item_Outlet_Sales)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", color = "blue", se = FALSE) +
  facet_wrap(~ Item_Type) +
  labs(title = "Item Visibility vs Outlet Sales by Item Type",
       x = "Item Visibility", y = "Outlet Sales") +
  theme_minimal()

# Creating scatter plots segmented by Outlet_Type
p_outlet_type <- ggplot(df, aes(x = Item_Visibility, y = Item_Outlet_Sales)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  facet_wrap(~ Outlet_Type) +
  labs(title = "Item Visibility vs Outlet Sales by Outlet Type",
       x = "Item Visibility", y = "Outlet Sales") +
  theme_minimal()

# Print the plots
print(p_item_type)
```

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

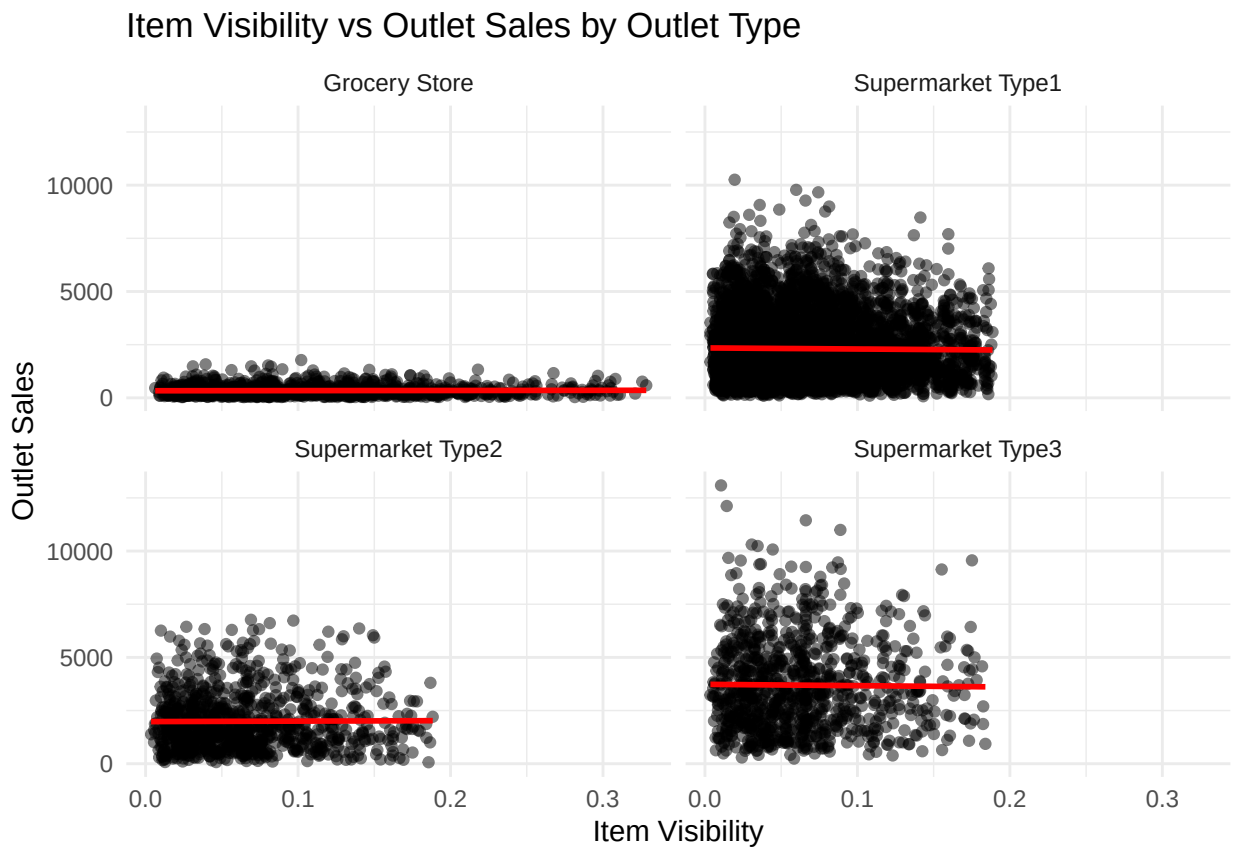
```
## Warning: Removed 115 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 115 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```
print(p_outlet_type)
```

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

## Warning: Removed 115 rows containing non-finite outside the scale range
## (`stat_smooth()`).
## Removed 115 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```r
# Fit a linear model
model <- lm(Item_Outlet_Sales ~ Item_Visibility + Item_Type + Outlet_Type + Item_Fat_Content + Item_MRP

# Summary of the model to understand influences
summary(model)
```

```
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility + Item_Type +
##     Outlet_Type + Item_Fat_Content + Item_MRP, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4326.1  -676.6   -86.1   568.2  7949.7
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -1860.3109    78.9138 -23.574   <2e-16 ***
## Item_Visibility                -229.6399   264.6602  -0.868    0.386
## Item_TypeBreads                  -0.9941    84.7102  -0.012    0.991
## Item_TypeBreakfast                4.9638   116.6266   0.043    0.966
## Item_TypeCanned                  10.2521    63.0520   0.163    0.871
## Item_TypeDairy                  -47.1681    62.5393  -0.754    0.451
## Item_TypeFrozen Foods           -31.5529    59.1922  -0.533    0.594
## Item_TypeFruits and Vegetables   24.4787    55.1999   0.443    0.657
## Item_TypeHard Drinks             10.0001    90.9802   0.110    0.912
```

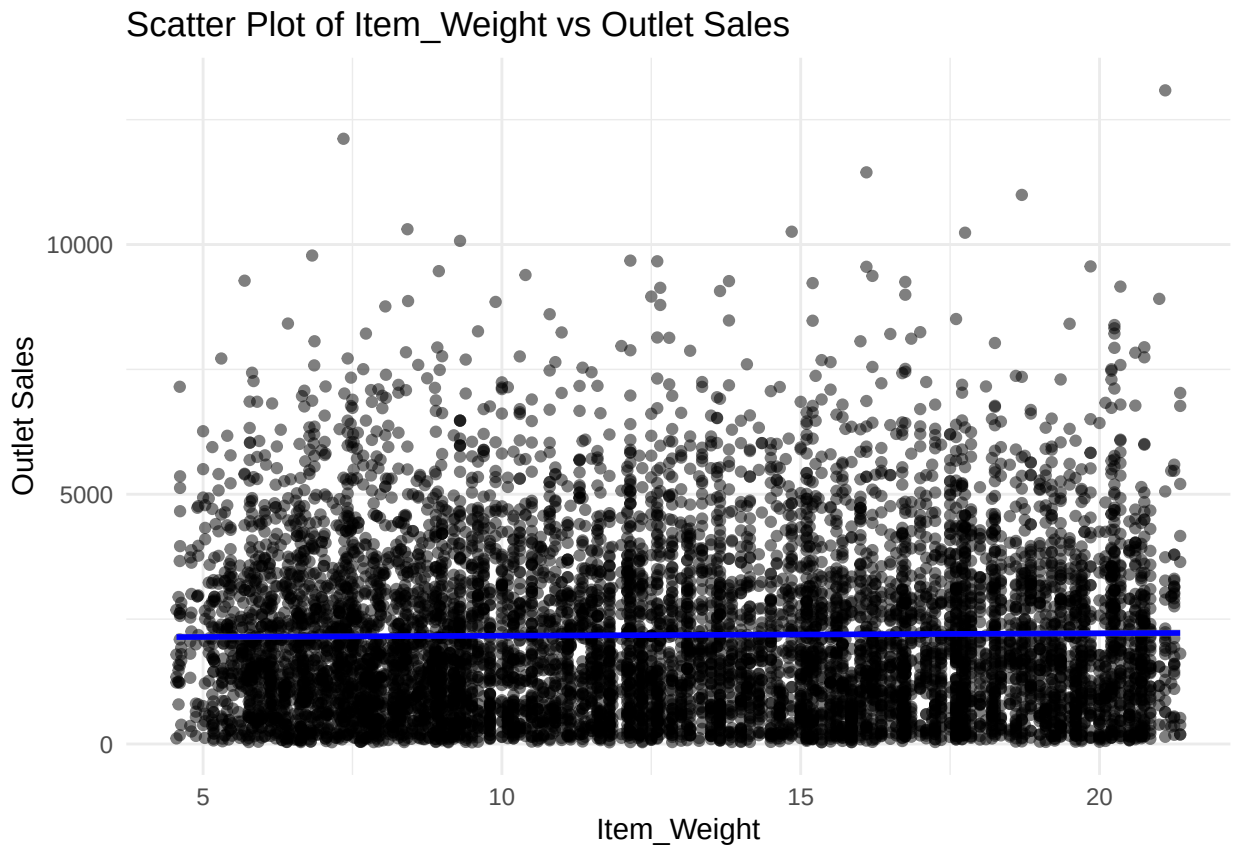
```
## Item_TypeHealth and Hygiene      -11.1540    68.3118  -0.163    0.870
## Item_TypeHousehold               -45.2971    60.2387  -0.752    0.452
## Item_TypeMeat                     -2.7989    71.1393  -0.039    0.969
## Item_TypeOthers                  -30.7078    98.8457  -0.311    0.756
## Item_TypeSeafood                 148.9574   150.1514   0.992    0.321
## Item_TypeSnack Foods             -15.4125    55.4897  -0.278    0.781
## Item_TypeSoft Drinks             -19.2240    70.7411  -0.272    0.786
## Item_TypeStarchy Foods             7.6562   103.8894   0.074    0.941
## Outlet_TypeSupermarket Type1    1949.2427    39.3623  49.521   <2e-16 ***
## Outlet_TypeSupermarket Type2    1623.2114    51.8204  31.324   <2e-16 ***
## Outlet_TypeSupermarket Type3    3350.0343    51.8830  64.569   <2e-16 ***
## Item_Fat_Content                  39.7699    28.3996   1.400    0.161
## Item_MRP                          15.5556     0.1989  78.190   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1129 on 8386 degrees of freedom
##   (115 observations deleted due to missingness)
## Multiple R-squared:  0.5643, Adjusted R-squared:  0.5632
## F-statistic: 517.2 on 21 and 8386 DF,  p-value: < 2.2e-16
```

```r
# Load necessary libraries
library(ggplot2)
library(dplyr)

# Convert categorical variables to factor if not already
df$Item_Type <- as.factor(df$Item_Type)
df$Outlet_Type <- as.factor(df$Outlet_Type)
df$Item_Fat_Content <- as.factor(df$Item_Fat_Content)
df$Outlet_Size <- as.factor(df$Outlet_Size)
df$Outlet_Location_Type <- as.factor(df$Outlet_Location_Type)

# Continuous Variables
continuous_vars <- c("Item_Weight", "Item_Visibility", "Item_MRP")

# Categorical Variables
categorical_vars <- c("Item_Type", "Outlet_Type", "Item_Fat_Content", "Outlet_Size", "Outlet_Location_Ty

# Plotting Distribution of Sales for Continuous Variables
for(var in continuous_vars) {
  p <- ggplot(df, aes_string(x = var, y = "Item_Outlet_Sales")) +
    geom_point(alpha = 0.5) +
    geom_smooth(method = "lm", color = "blue") +
    labs(title = paste("Scatter Plot of", var, "vs Outlet Sales"),
         x = var, y = "Outlet Sales") +
    theme_minimal()
  print(p)
}
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
```

```
## generated.
```

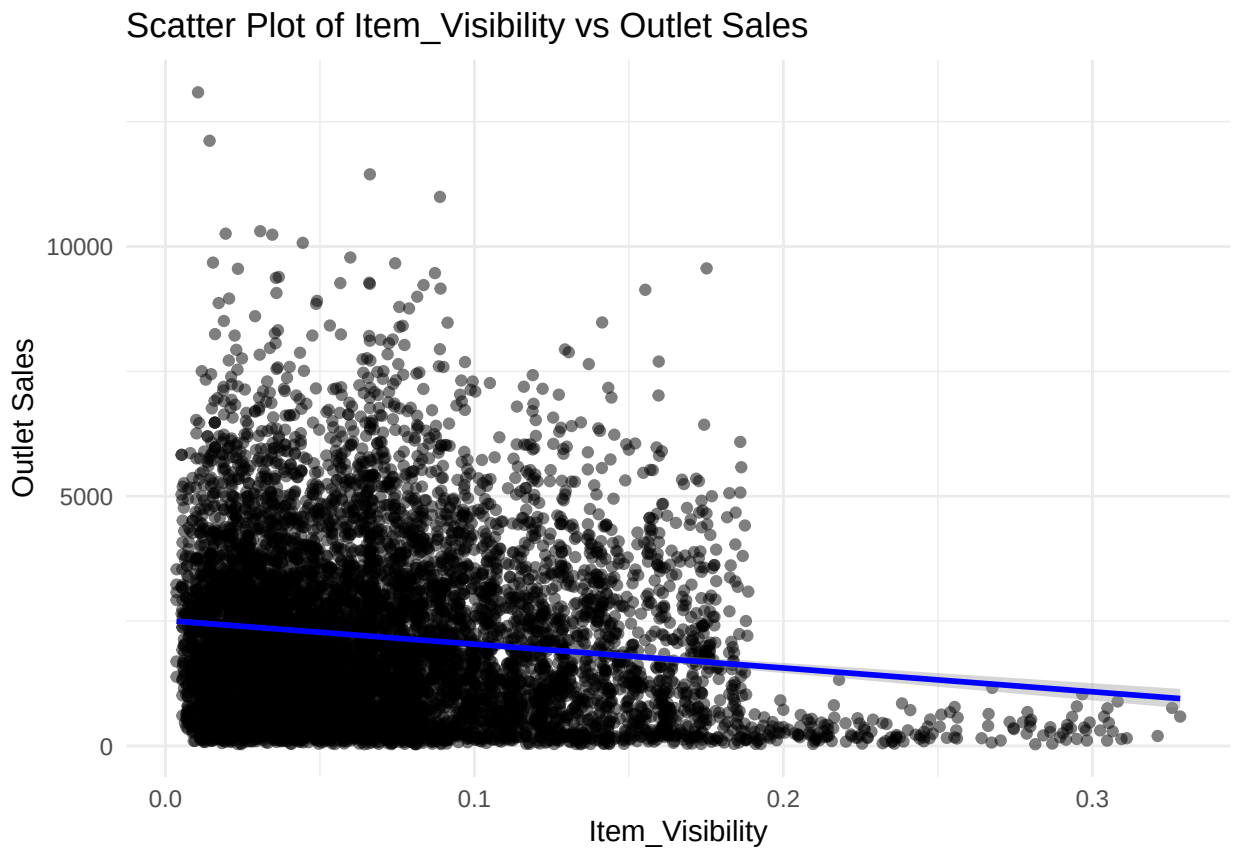
```
## `geom_smooth()` using formula = 'y ~ x'
```



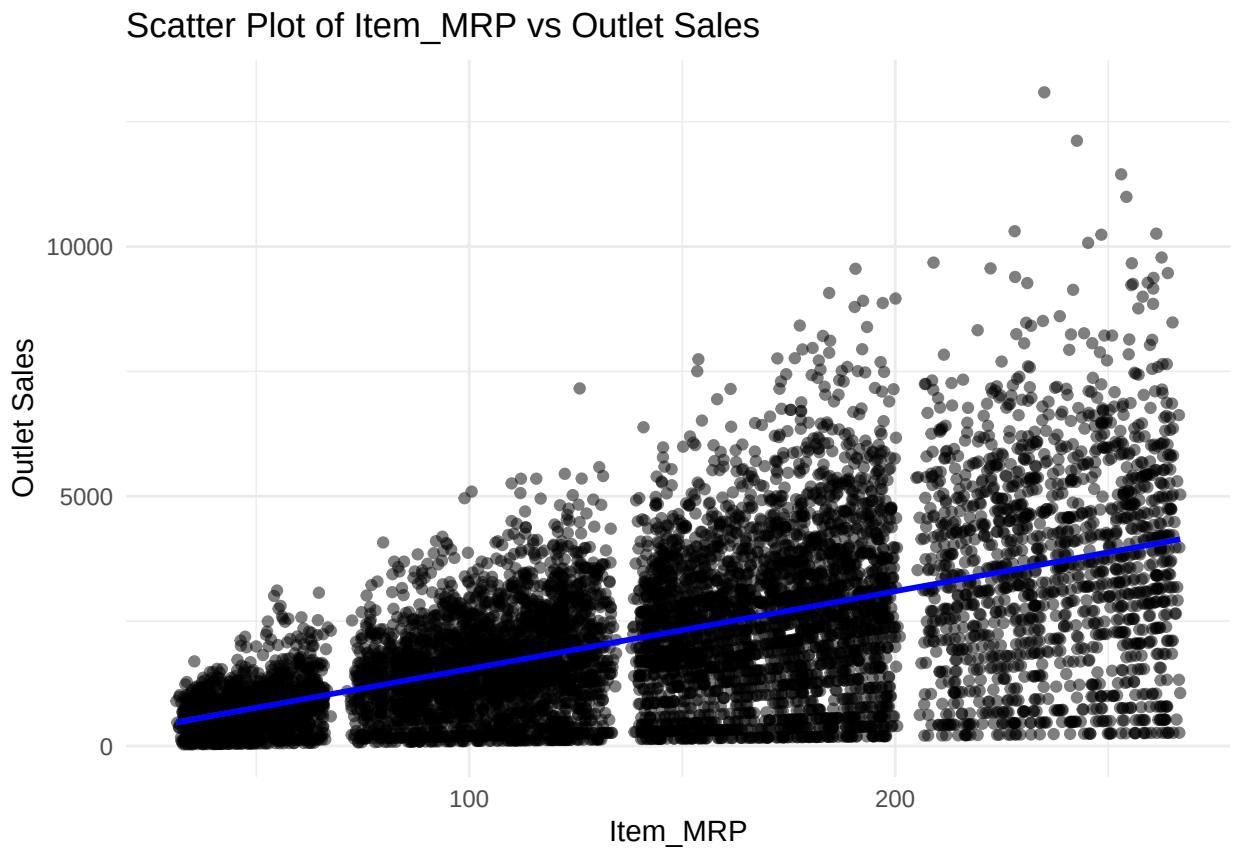

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 115 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 115 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



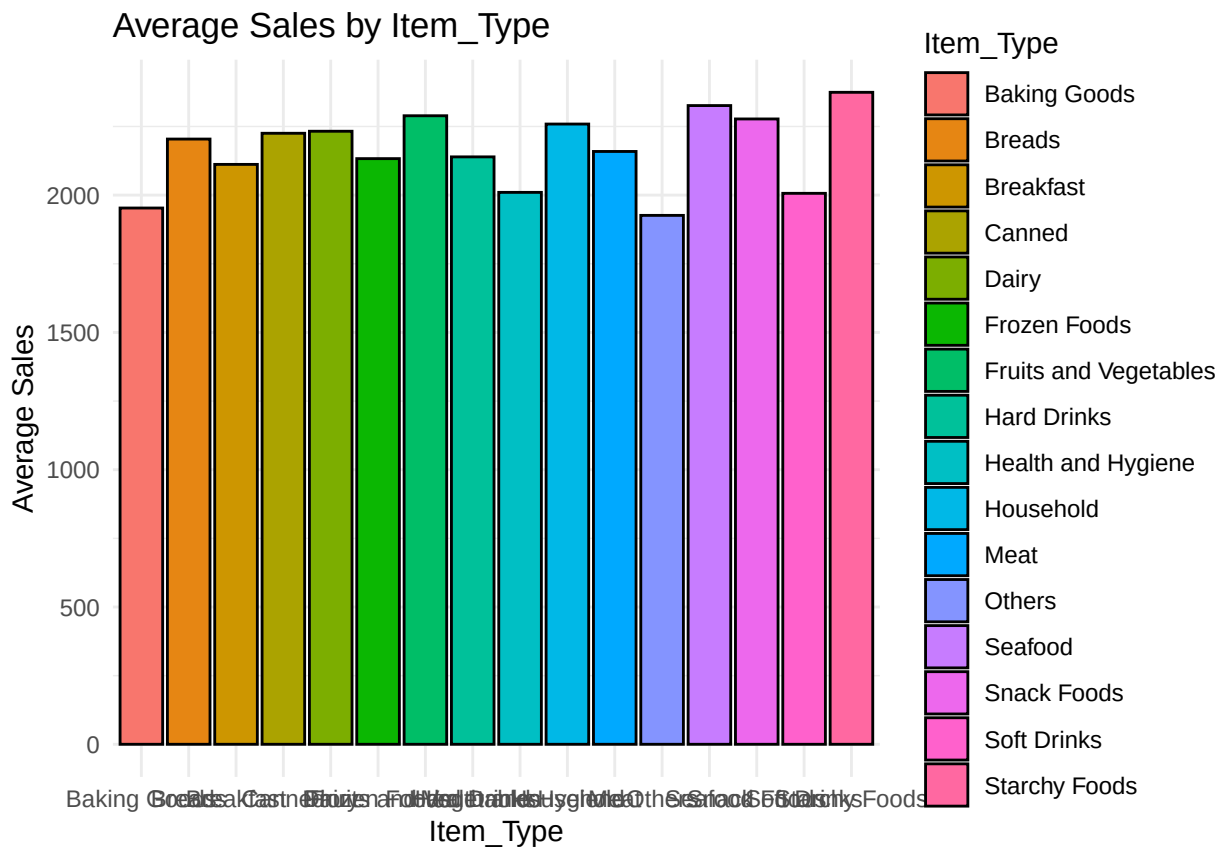

```
## `geom_smooth()` using formula = 'y ~ x'
```



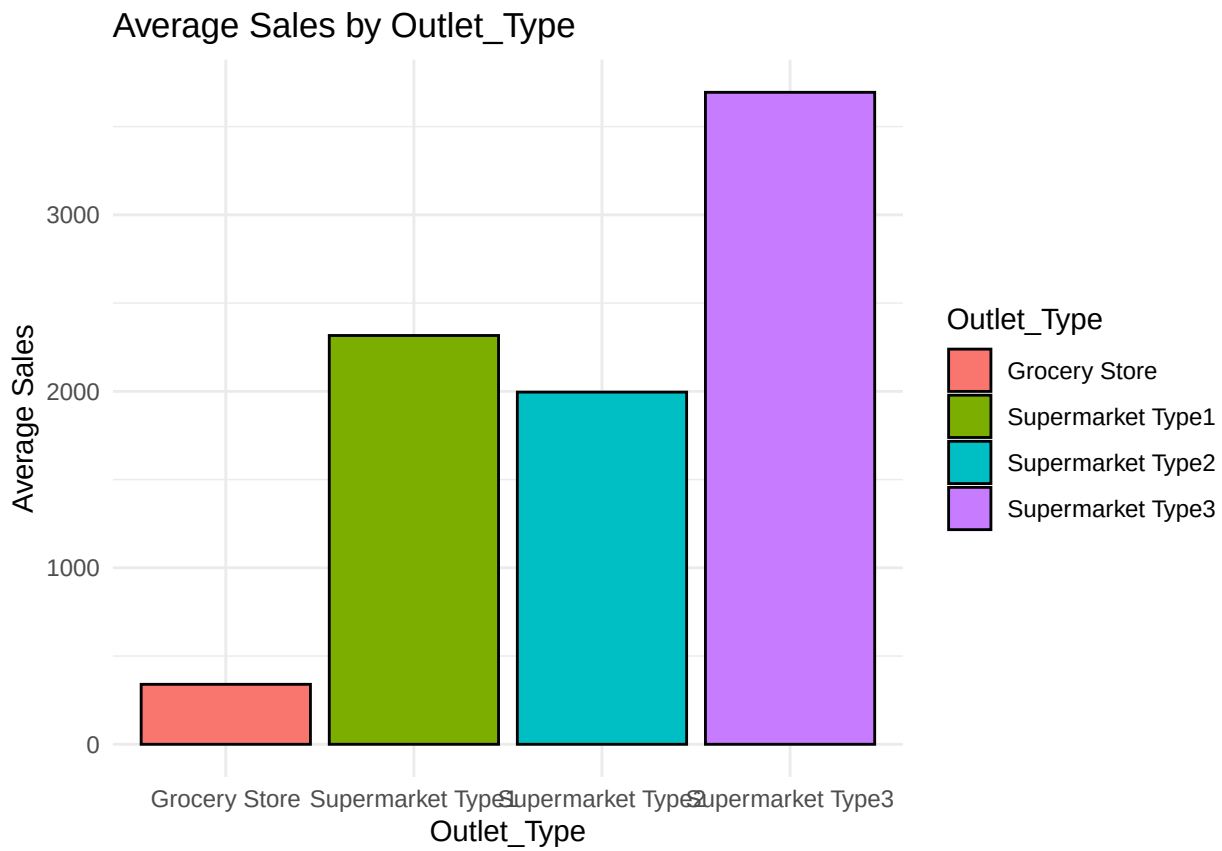

```
# Plotting Distribution of Sales for Categorical Variables
for(var in categorical_vars) {
  p <- df %>%
    group_by_(.dots = var) %>%
    summarise(Average_Sales = mean(Item_Outlet_Sales)) %>%
    ggplot(aes_string(x = var, y = "Average_Sales", fill = var)) +
    geom_bar(stat = "identity", color = "black") +
    labs(title = paste("Average Sales by", var),
         x = var, y = "Average Sales") +
    theme_minimal()
  print(p)
}
```

```
## Warning: `group_by_()` was deprecated in dplyr 0.7.0.
## i Please use `group_by()` instead.
## i See vignette('programming') for more help
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

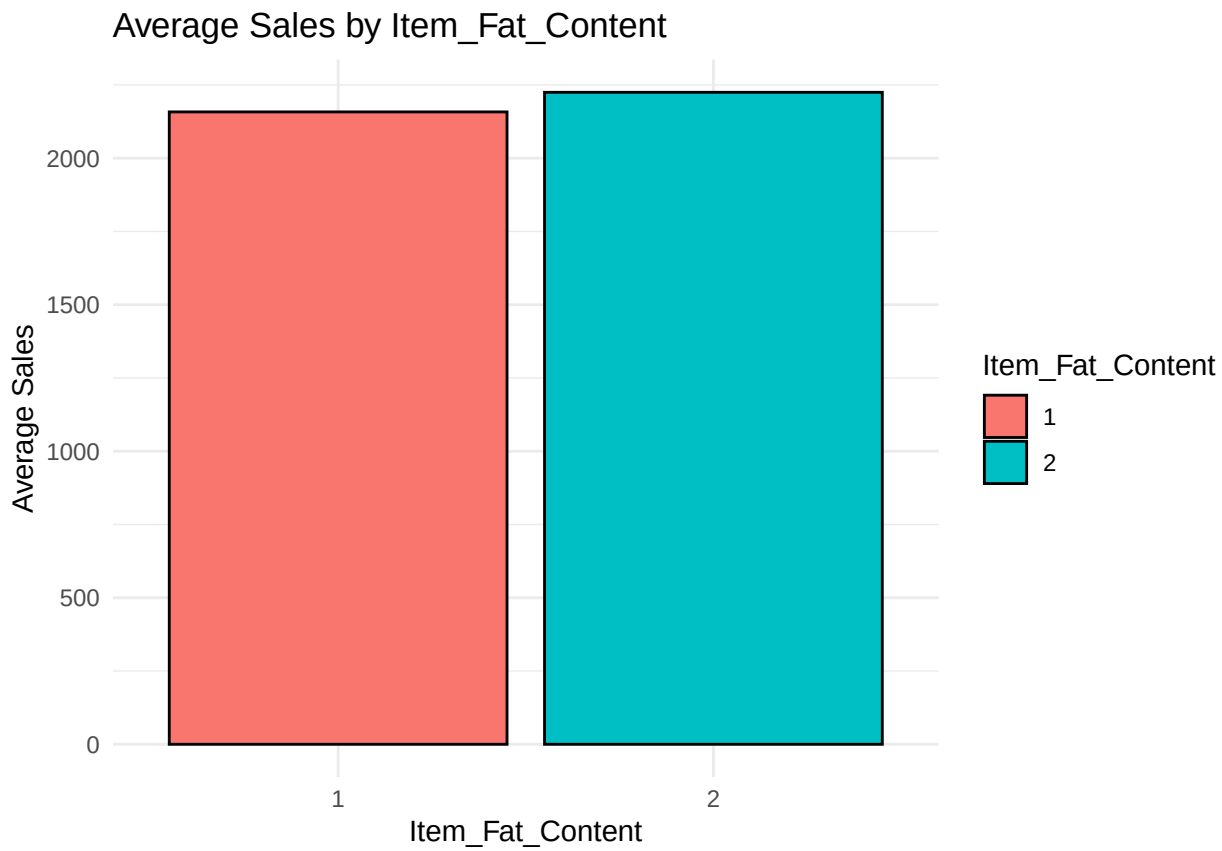
```
## Warning: `group_by_()` was deprecated in dplyr 0.7.0.
## i Please use `group_by()` instead.
## i See vignette('programming') for more help
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



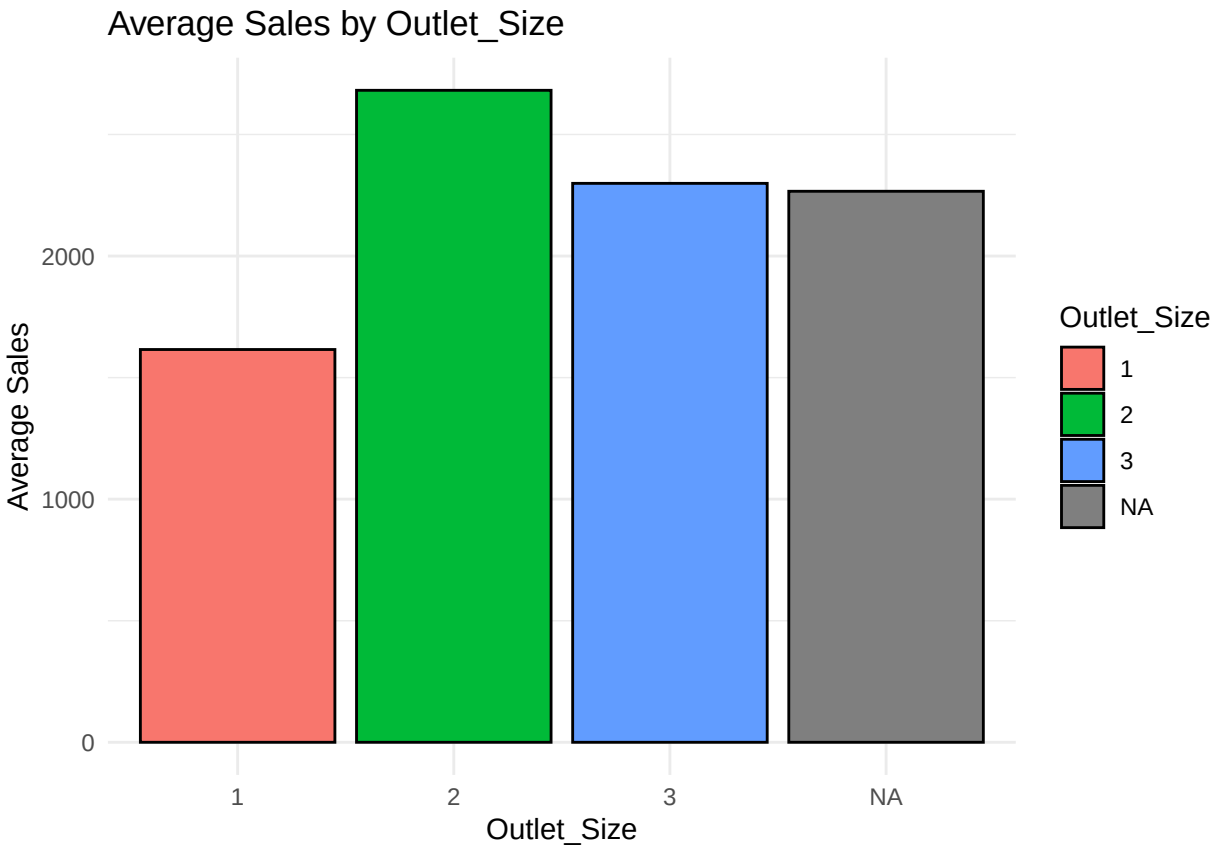

```
## Warning: `group_by_()` was deprecated in dplyr 0.7.0.
## i Please use `group_by()` instead.
## i See vignette('programming') for more help
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

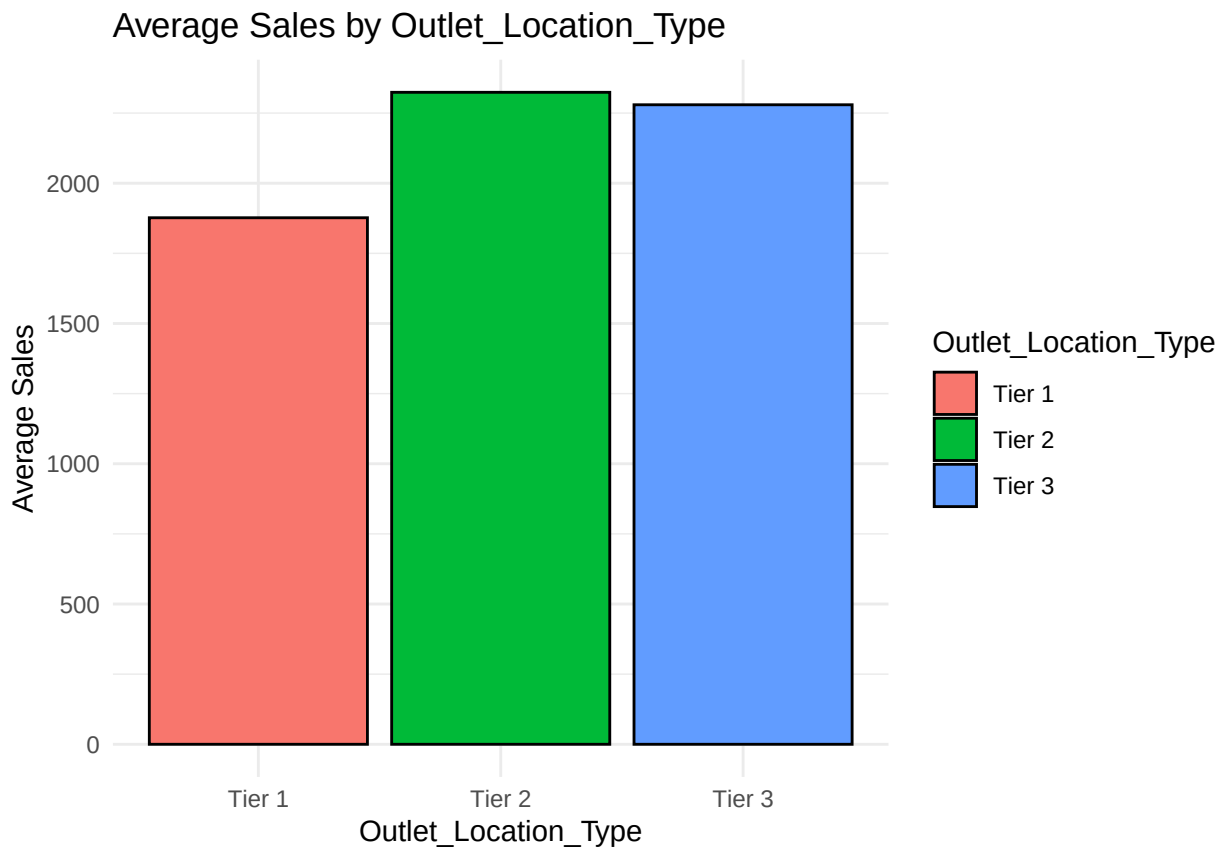



```
## Warning: `group_by_()` was deprecated in dplyr 0.7.0.
## i Please use `group_by()` instead.
## i See vignette('programming') for more help
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



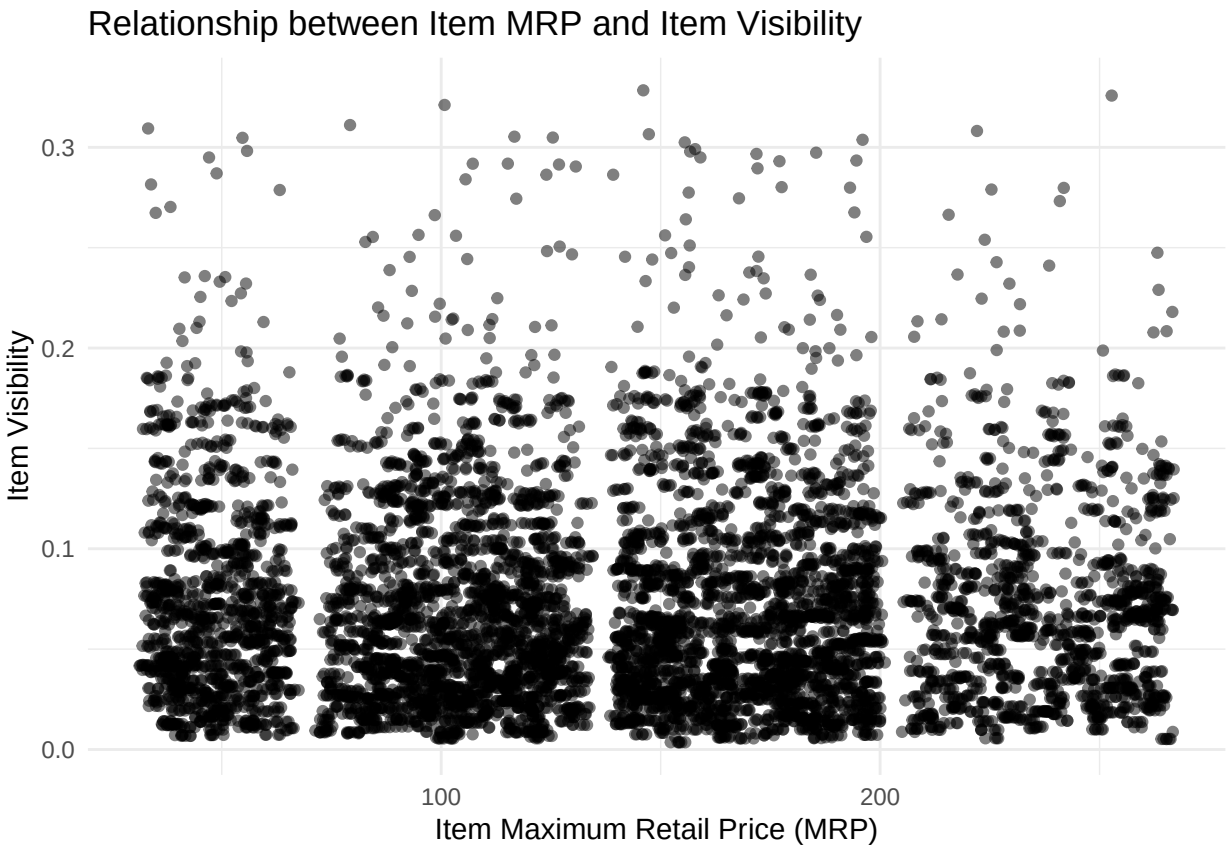

```
## Warning: ‘group_by_()‘ was deprecated in dplyr 0.7.0.
## i Please use ‘group_by()‘ instead.
## i See vignette(’programming’) for more help
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```


Average Sales by Outlet_Size
Average Sales
2000
1000
0
1
2
3
NA
Outlet_Size
Outlet_Size
1
2
3
NA

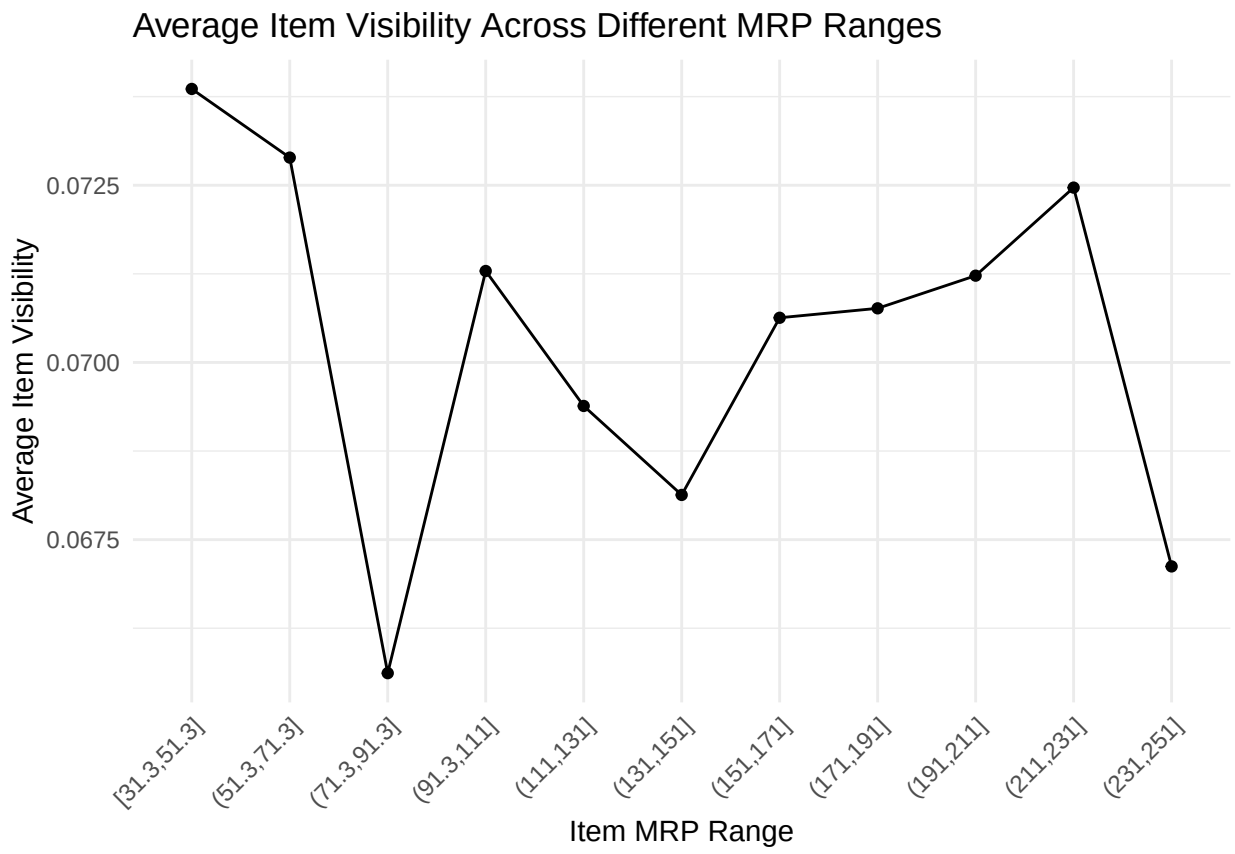



```
# Load ggplot2
library(ggplot2)


# Create a scatter plot
ggplot(df, aes(x = Item_MRP, y = Item_Visibility)) +
  geom_point(alpha = 0.5) +  # Use semi-transparent points to handle overplotting
  labs(x = "Item Maximum Retail Price (MRP)",
       y = "Item Visibility",
       title = "Relationship between Item MRP and Item Visibility") +
  theme_minimal()  # Clean minimalistic theme
```

```
## Warning: Removed 115 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```r
# Create bins for Item MRP using a reasonable interval
df$MRP_Bin <- cut(df$Item_MRP, breaks=seq(from=min(df$Item_MRP), to=max(df$Item_MRP), by=20), include.l

# Calculate average visibility per MRP bin
average_visibility_per_mrp <- aggregate(Item_Visibility ~ MRP_Bin, data = df, mean)

# Create a line graph
ggplot(average_visibility_per_mrp, aes(x = MRP_Bin, y = Item_Visibility, group=1)) +
  geom_line() +  # Adds a line graph
  geom_point() +  # Adds points to each average point
  labs(x = "Item MRP Range", y = "Average Item Visibility",
       title = "Average Item Visibility Across Different MRP Ranges") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))  # Improve x-axis label readability
```



gli item che fanno vedere di può sono quelli che vendono a di meno quindi quelli di cui vogliono sbarazzarsi.

```
# Load necessary libraries
library(ggplot2)
library(dplyr)




# Segmented Regression by Outlet Type
outlet_type_models <- df %>%
  group_by(Outlet_Type) %>%
  do(model = lm(Item_Outlet_Sales ~ Item_Visibility, data = .))

# Viewing summaries for each Outlet Type
outlet_type_summaries <- lapply(outlet_type_models$model, summary)

# Print the summaries for review
print(outlet_type_summaries)
```

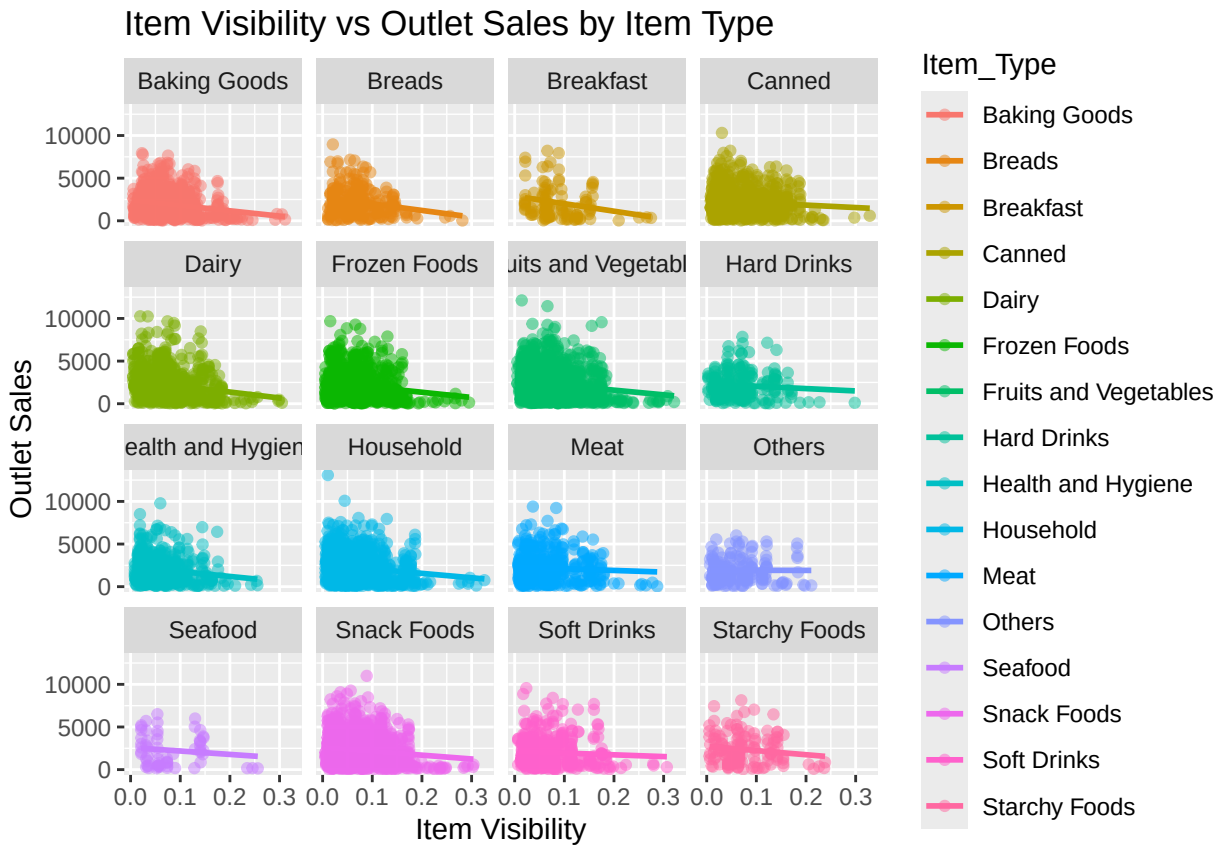
```
## [[1]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
```

```
##    Min     1Q Median     3Q    Max
## -315.9 -185.5  -82.7  119.4 1436.3
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       332.59      14.53  22.884   <2e-16 ***
## Item_Visibility    66.22     111.44   0.594    0.553
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 260.9 on 1081 degrees of freedom
## Multiple R-squared:  0.0003265,  Adjusted R-squared:  -0.0005983
## F-statistic: 0.353 on 1 and 1081 DF,  p-value: 0.5525
##
##
## [[2]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2229.2 -1161.7  -317.7   817.1  7919.3
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2347.26      37.59  62.440   <2e-16 ***
## Item_Visibility  -509.99     484.61  -1.052    0.293
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1514 on 5460 degrees of freedom
##   (115 observations deleted due to missingness)
## Multiple R-squared:  0.0002028,  Adjusted R-squared:  1.968e-05
## F-statistic: 1.107 on 1 and 5460 DF,  p-value: 0.2927
##
##
## [[3]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1953.2 -1017.5  -346.2   711.6  4772.2
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      1980.78      83.49   23.73   <2e-16 ***
## Item_Visibility   224.60    1071.52    0.21    0.834
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1377 on 926 degrees of freedom
```

```
## Multiple R-squared:  4.744e-05,  Adjusted R-squared:  -0.001032
## F-statistic: 0.04393 on 1 and 926 DF,  p-value: 0.834
##
##
## [[4]]
##
## Call:
## lm(formula = Item_Outlet_Sales ~ Item_Visibility, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3455.1 -1654.9  -342.8  1274.6  9360.0
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       3733.7      128.7  29.003   <2e-16 ***
## Item_Visibility   -632.0     1726.5  -0.366    0.714
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2129 on 933 degrees of freedom
## Multiple R-squared:  0.0001436,  Adjusted R-squared:  -0.0009281
## F-statistic: 0.134 on 1 and 933 DF,  p-value: 0.7144
```

```r
# Optional: Plotting the regression lines for each type on a scatter plot
# Plotting for Item Type
ggplot(df, aes(x = Item_Visibility, y = Item_Outlet_Sales, color = Item_Type)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", se = FALSE) +
  facet_wrap(~ Item_Type) +
  labs(title = "Item Visibility vs Outlet Sales by Item Type", x = "Item Visibility", y = "Outlet Sales
```

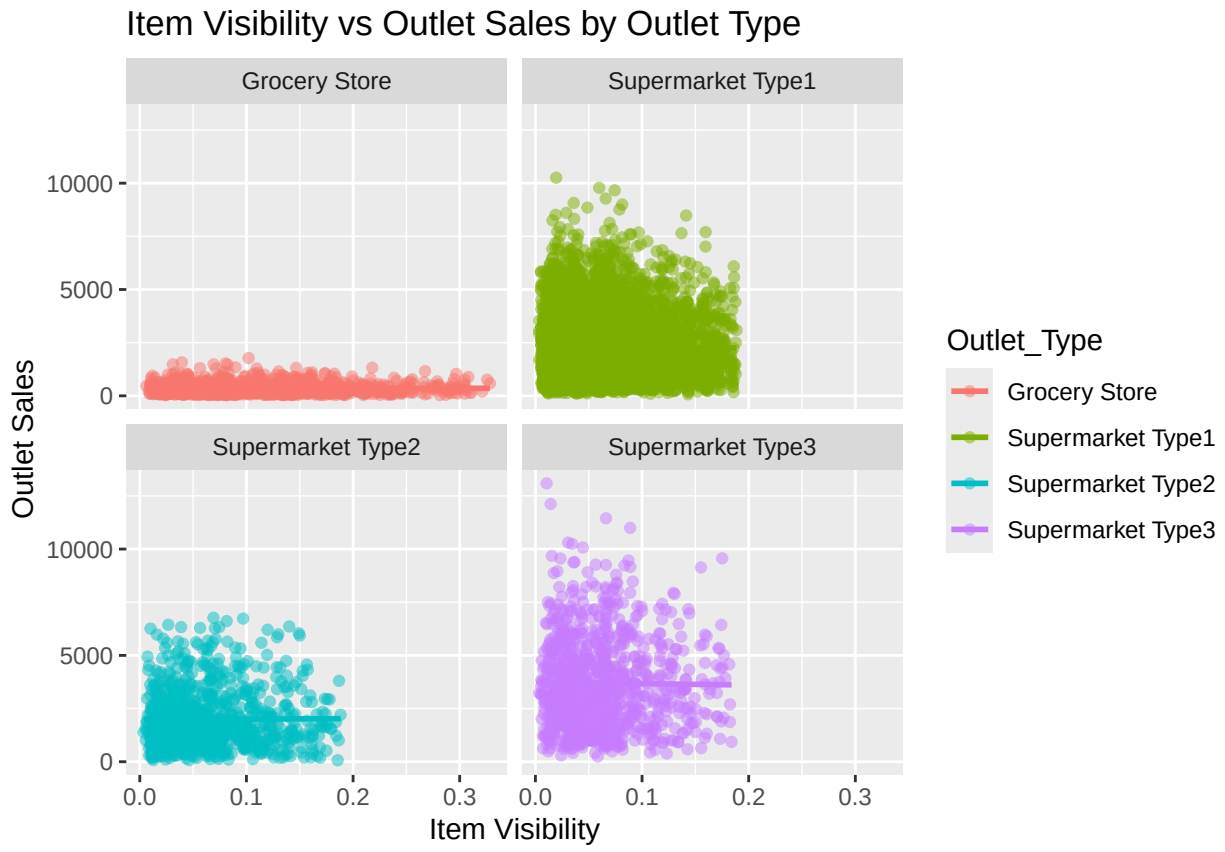
```
## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 115 rows containing non-finite outside the scale range
## (`stat_smooth()`).

## Warning: Removed 115 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```
# Plotting for Outlet Type
ggplot(df, aes(x = Item_Visibility, y = Item_Outlet_Sales, color = Outlet_Type)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", se = FALSE) +
  facet_wrap(~ Outlet_Type) +
  labs(title = "Item Visibility vs Outlet Sales by Outlet Type", x = "Item Visibility", y = "Outlet Sal
```

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

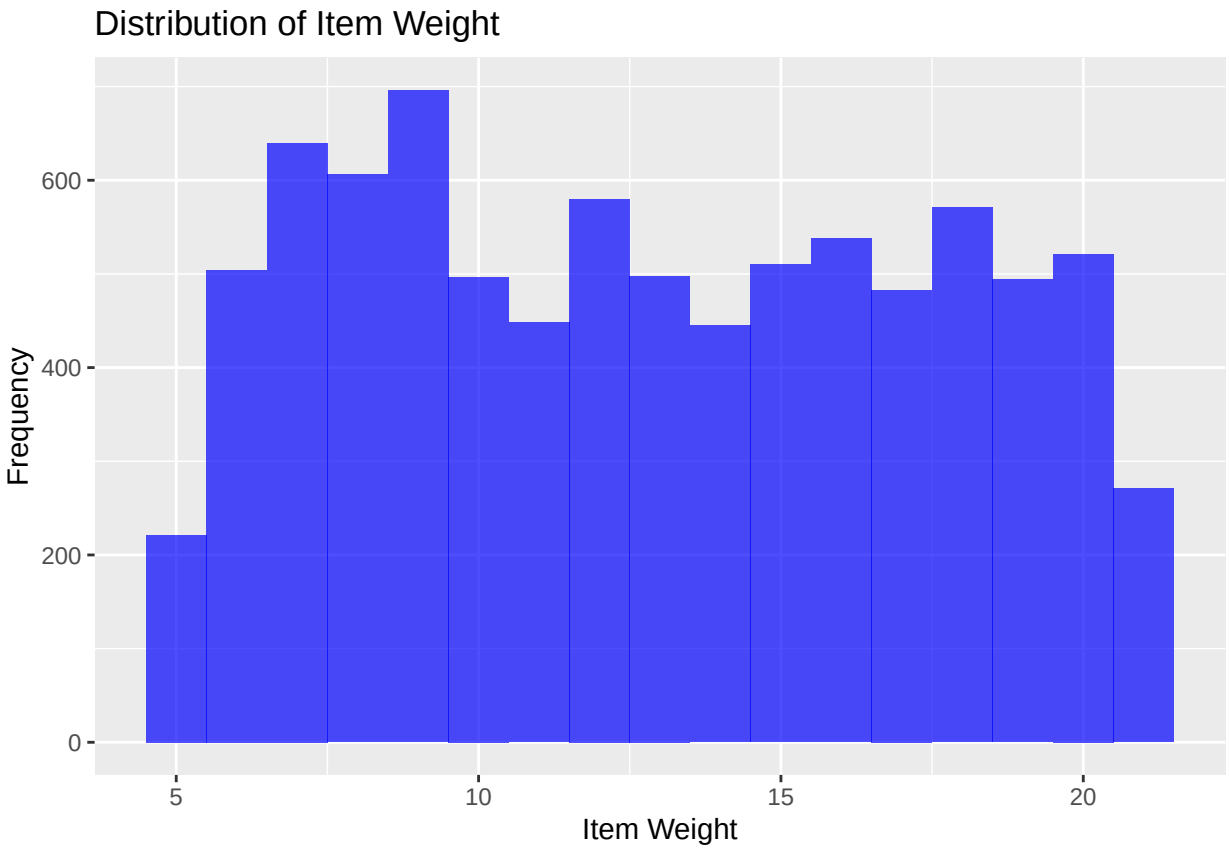
```
## Warning: Removed 115 rows containing non-finite outside the scale range
## (`stat_smooth()`).
## Removed 115 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



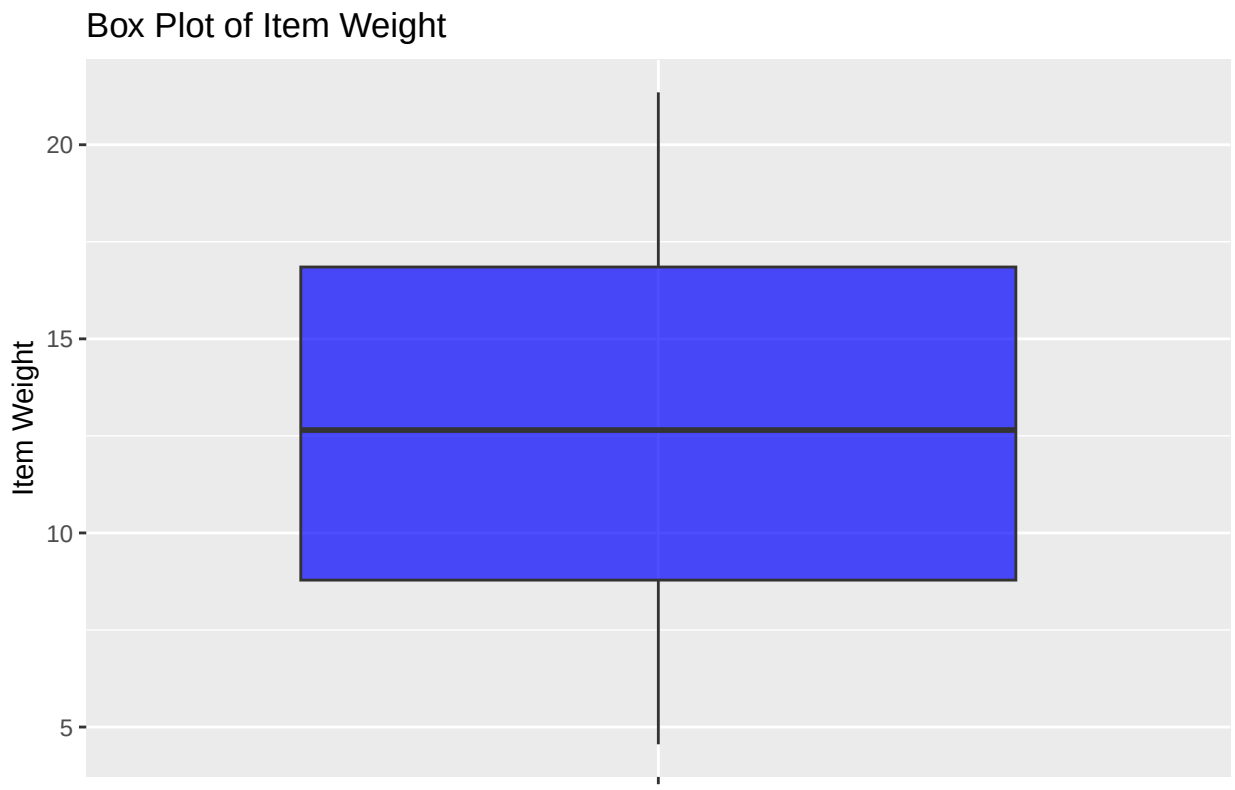
###Checking for Outliers

```
# Load necessary libraries
library(ggplot2)
library(gridExtra)

# Histogram and Box Plot for Item_Weight
p1 <- ggplot(df, aes(x = Item_Weight)) +
  geom_histogram(binwidth = 1, fill = 'blue', alpha = 0.7) +
  ggtitle("Distribution of Item Weight") +
  xlab("Item Weight") +
  ylab("Frequency")

p2 <- ggplot(df, aes(x = "", y = Item_Weight)) +
  geom_boxplot(fill = 'blue', alpha = 0.7) +
  ggtitle("Box Plot of Item Weight") +
  xlab("") +
  ylab("Item Weight")

# Histogram and Box Plot for Item_Visibility
p3 <- ggplot(df, aes(x = Item_Visibility)) +
  geom_histogram(binwidth = 0.01, fill = 'green', alpha = 0.7) +
  ggtitle("Distribution of Item Visibility") +
  xlab("Item Visibility") +
  ylab("Frequency")

p4 <- ggplot(df, aes(x = "", y = Item_Visibility)) +
```

```
  geom_boxplot(fill = 'green', alpha = 0.7) +
  ggtitle("Box Plot of Item Visibility") +
  xlab("") +
  ylab("Item Visibility")

# Histogram and Box Plot for Item_MRP
p5 <- ggplot(df, aes(x = Item_MRP)) +
  geom_histogram(binwidth = 5, fill = 'red', alpha = 0.7) +
  ggtitle("Distribution of Item MRP") +
  xlab("Item MRP") +
  ylab("Frequency")

p6 <- ggplot(df, aes(x = "", y = Item_MRP)) +
  geom_boxplot(fill = 'red', alpha = 0.7) +
  ggtitle("Box Plot of Item MRP") +
  xlab("") +
  ylab("Item MRP")

# Histogram and Box Plot for Item_Outlet_Sales
p7 <- ggplot(df, aes(x = Item_Outlet_Sales)) +
  geom_histogram(binwidth = 100, fill = 'purple', alpha = 0.7) +
  ggtitle("Distribution of Item Outlet Sales") +
  xlab("Item Outlet Sales") +
  ylab("Frequency")

p8 <- ggplot(df, aes(x = "", y = Item_Outlet_Sales)) +
  geom_boxplot(fill = 'purple', alpha = 0.7) +
  ggtitle("Box Plot of Item Outlet Sales") +
  xlab("") +
  ylab("Item Outlet Sales")

# Print the plots individually
print(p1)
```

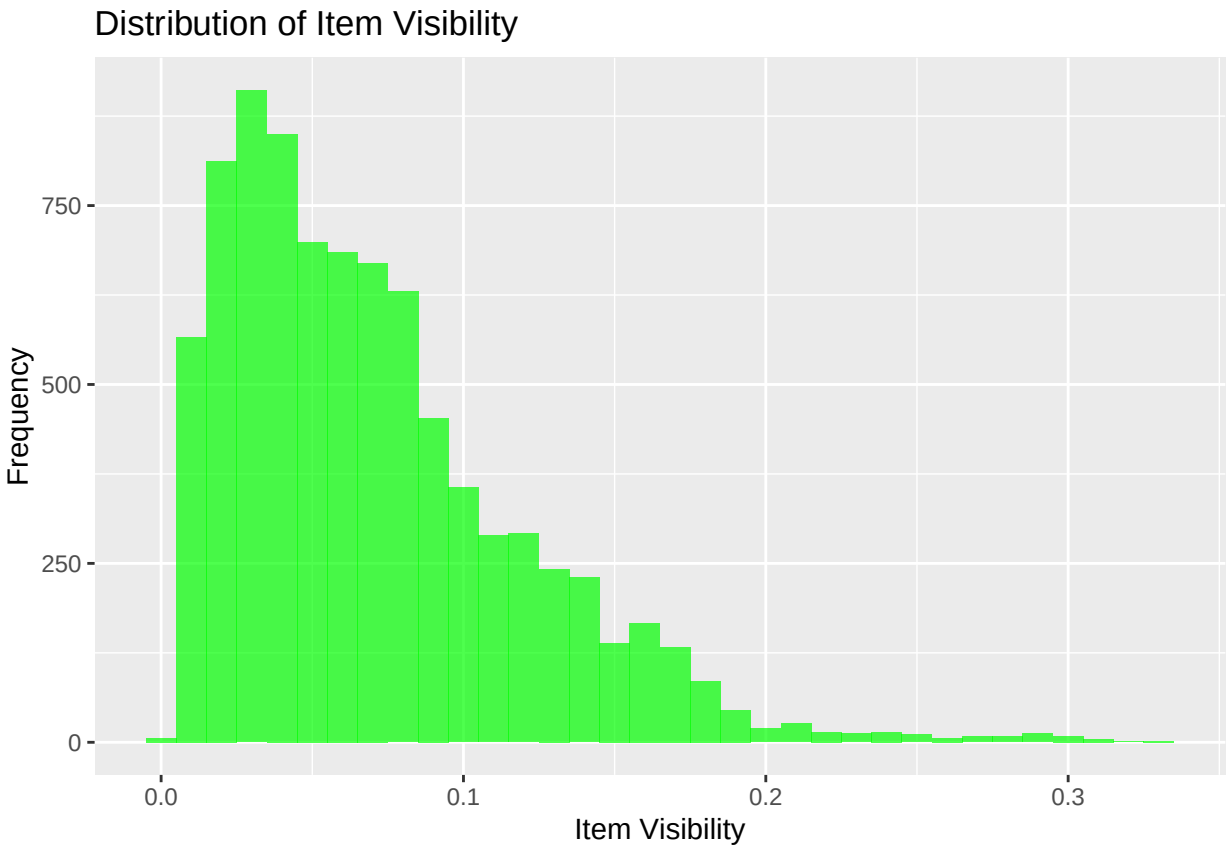

Distribution of Item Weight
Frequency
600
400
200
0
5
10
15
20
Item Weight

```
print(p2)
```



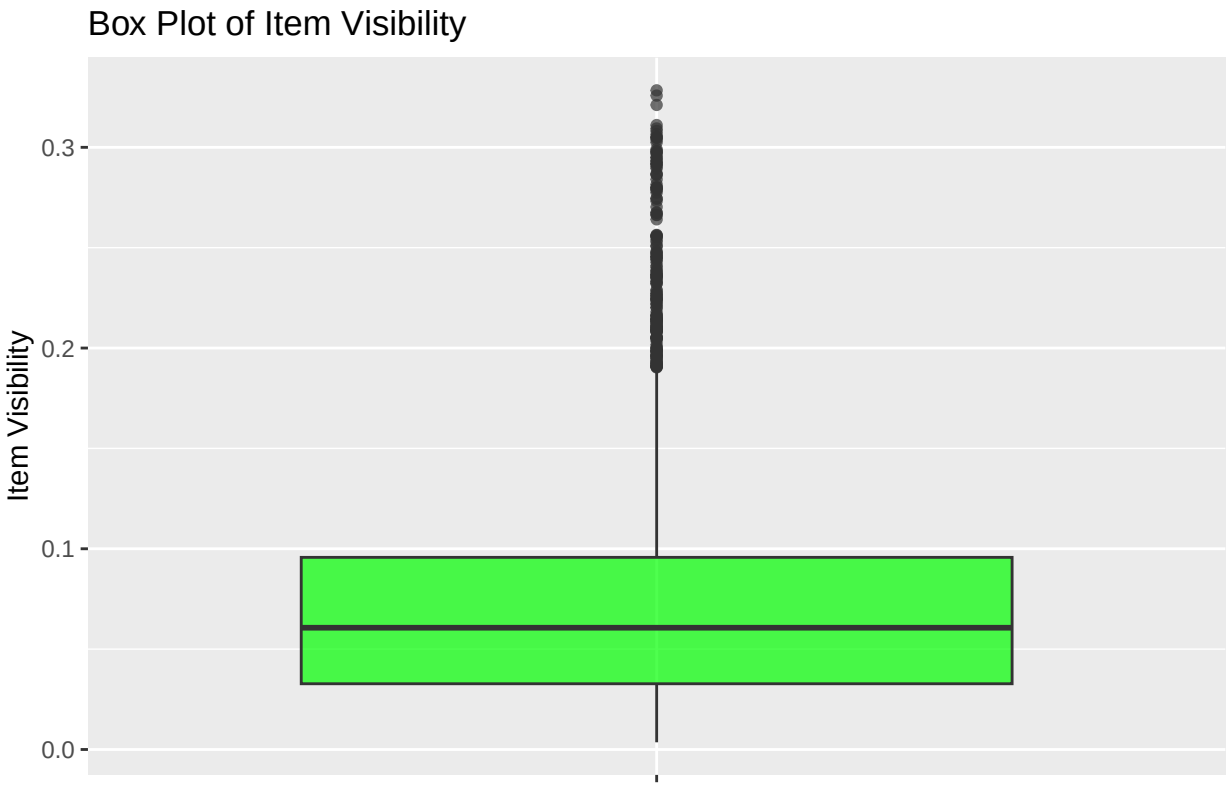

```
print(p3)
```

```
## Warning: Removed 115 rows containing non-finite outside the scale range
## (‘stat_bin()‘).
```

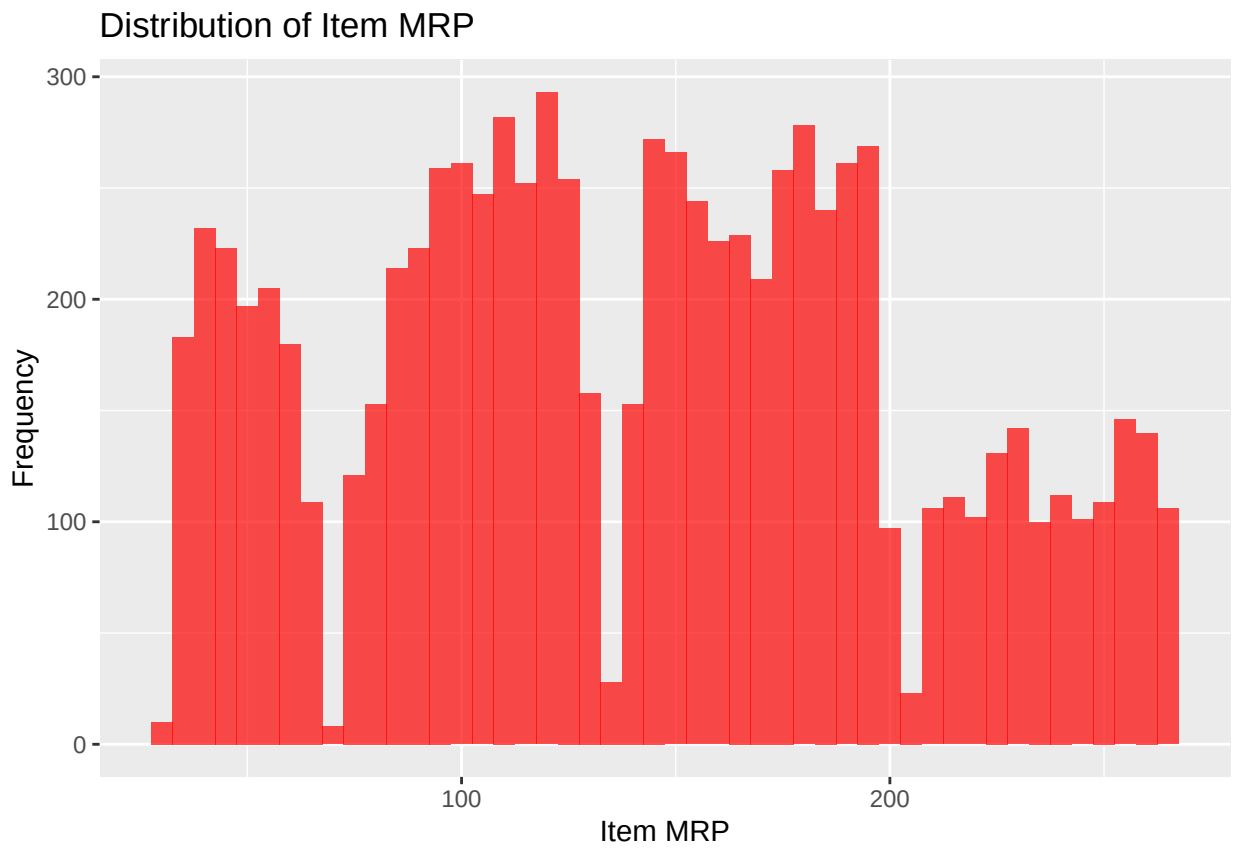



```
print(p4)
```

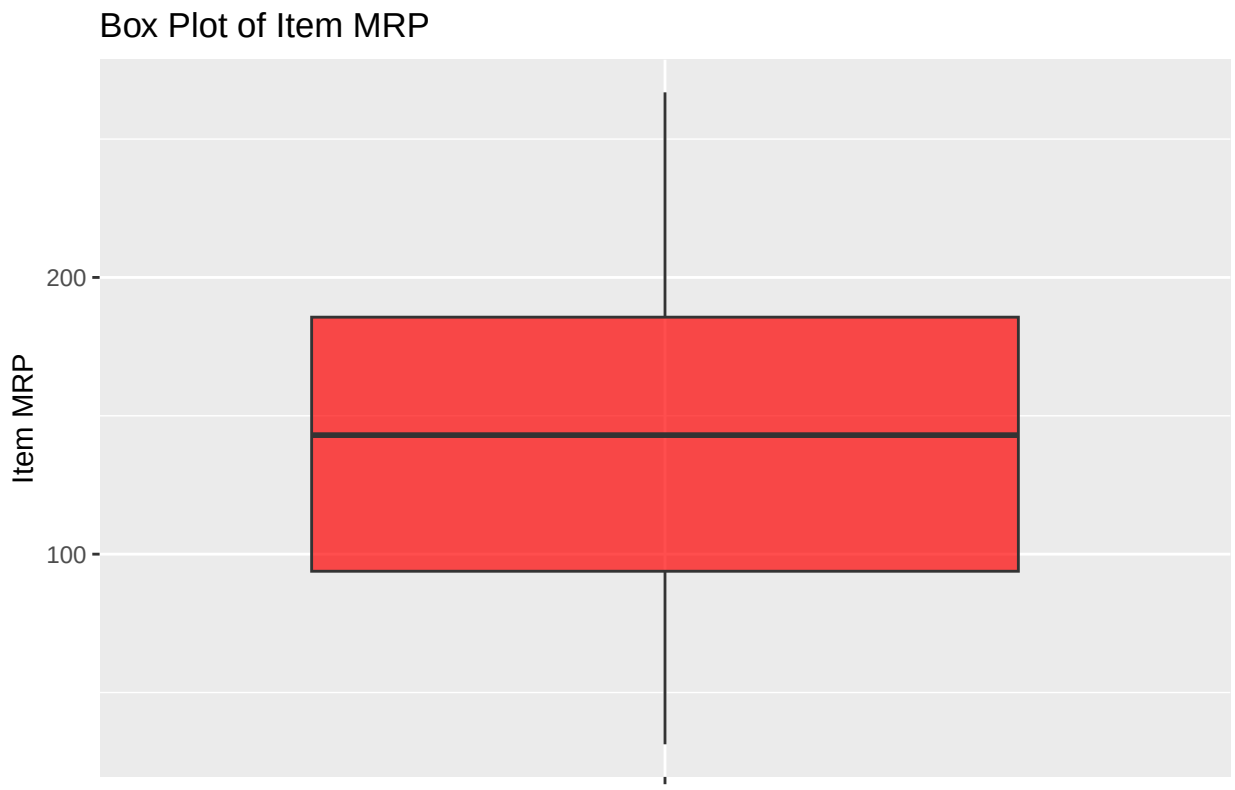
```
## Warning: Removed 115 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```



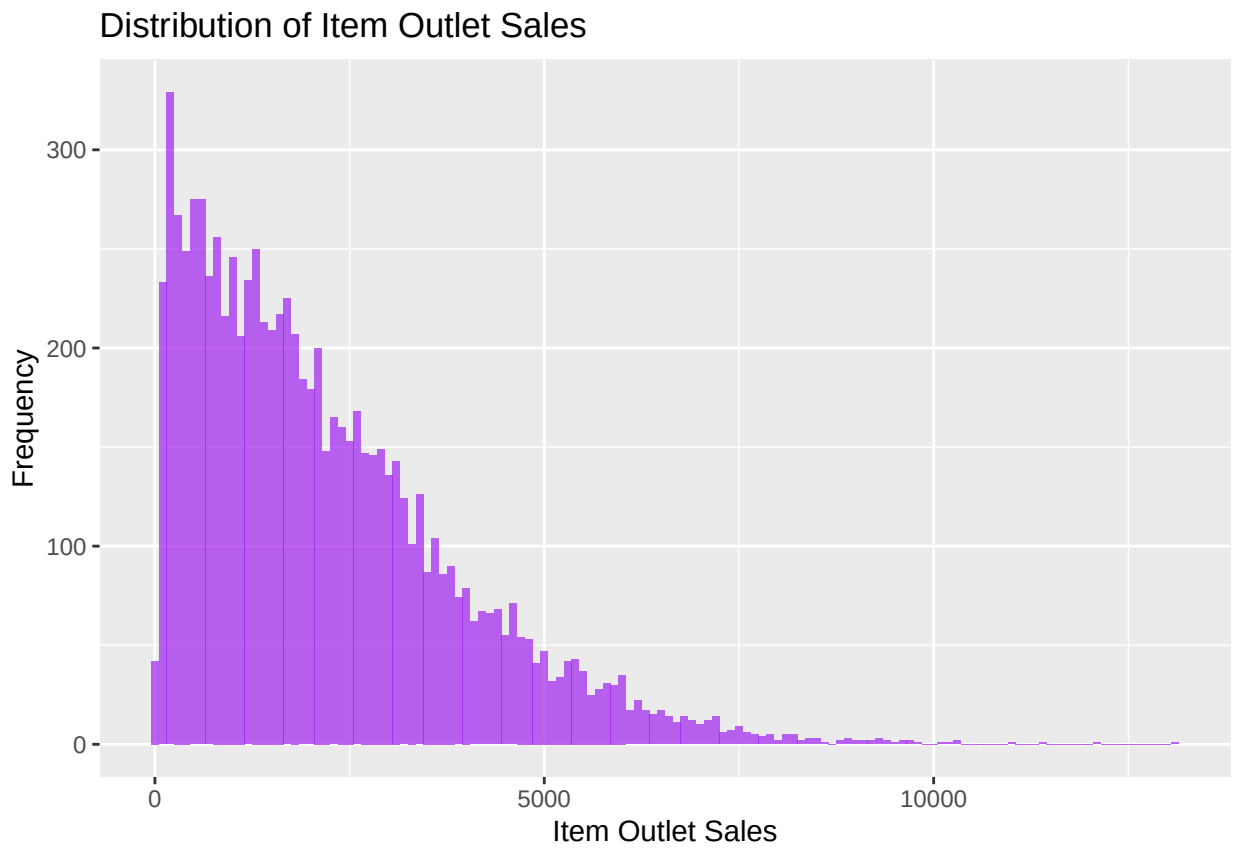

```
print(p5)
```


Distribution of Item MRP
300
200
100
0
Frequency
100
200
Item MRP

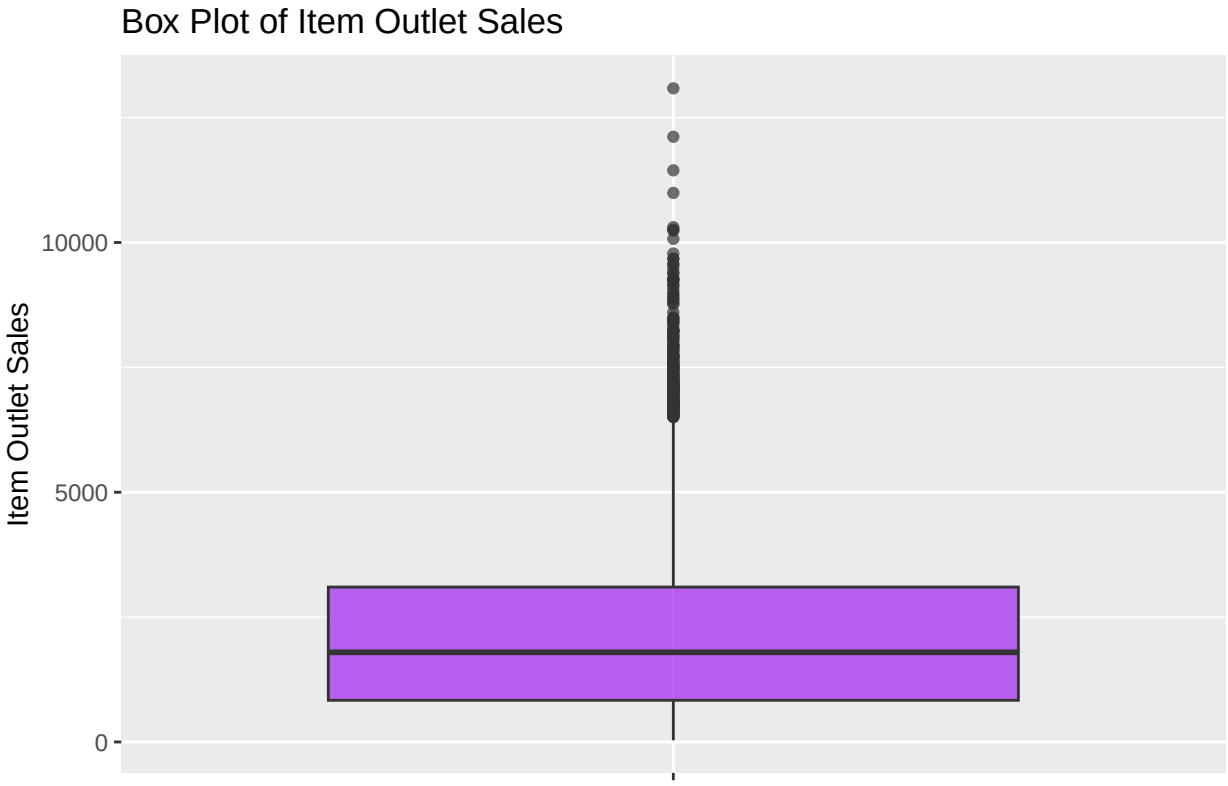
```
print(p6)
```




```
print(p7)
```


Distribution of Item Outlet Sales
Frequency
300
200
100
0
0
5000
10000
Item Outlet Sales

```
print(p8)
```


Box Plot of Item Outlet Sales
Item Outlet Sales
10000
5000
0