

SALES ANALYSIS

Faleke Micheal Kayode

2026-02-26

```
##importing the sales dataset into R
```

```
pizza_sales <- read.csv("~/pizza_sales.csv")
```

```
##installing neccessary functions into r
```

```
library(tidyverse)
```

```
## — Attaching core tidyverse packages — tidyverse 2.0.0 —
```

```
## ✓ dplyr 1.1.4 ✓ readr 2.1.5
## ✓ forcats 1.0.0 ✓ stringr 1.6.0
## ✓ ggplot2 4.0.2 ✓ tibble 3.3.0
## ✓ lubridate 1.9.4 ✓ tidyr 1.3.1
## ✓ purrr 1.1.0
```

```
## — Conflicts — tidyverse_conflicts() —
```

```
## ✖ dplyr::filter() masks stats::filter()
```

```
## ✖ dplyr::lag() masks stats::lag()
```

```
## ⓘ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(stringr)
```

```
library(lubridate)
```

```
library(ggplot2)
```

```
##inspecting the dataset
```

```
glimpse(pizza_sales)
```

```
## Rows: 48,620
```

```
## Columns: 12
```

```
## $ pizza_id <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 1...
```

```
## $ order_id <int> 1, 2, 2, 2, 2, 2, 3, 3, 4, 5, 6, 6, 7, 8, 9, 9, 9, 9...
```

```
## $ pizza_name_id <chr> "hawaiian_m", "classic_dlx_m", "five_cheese_l", "ita...
```

```
## $ quantity <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
```

```
## $ order_date <chr> "1/1/2015", "1/1/2015", "1/1/2015", "1/1/2015", "1/1...
```

```
## $ order_time <chr> "11:38:36 AM", "11:57:40 AM", "11:57:40 AM", "11:57:...
```

```
## $ unit_price <dbl> 13.25, 16.00, 18.50, 20.75, 16.00, 20.75, 16.50, 20....
```

```
## $ total_price <dbl> 13.25, 16.00, 18.50, 20.75, 16.00, 20.75, 16.50, 20....
```

```
## $ pizza_size <chr> "M", "M", "L", "L", "M", "L", "M", "L", "M", "M", "S...
```

```
## $ pizza_category <chr> "Classic", "Classic", "Veggie", "Supreme", "Veggie",...
```

```
## $ pizza_ingredients <chr> "Sliced Ham, Pineapple, Mozzarella Cheese", "Pepper...
```

```
## $ pizza_name <chr> "The Hawaiian Pizza", "The Classic Deluxe Pizza", "T...
```

```
##inspecting the dataset to see if there are any missing values
```

```
any(is.na(pizza_sales))
```

```
## [1] FALSE
```

```
##inspecting the rows of the dataset to see if there are any missing values
```

```
rowSums(is.na(pizza_sales))
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

##inspecting the columns in the dataset to see if there are any missing values

```
##      pizza_id      order_id      pizza_name_id      quantity
##      0            0            0            0
##      order_date      order_time      unit_price      total_price
##      0            0            0            0
##      pizza_size      pizza_category      pizza_ingredients      pizza_name
##      0            0            0            0
```

```
Pizza_sales<- pizza_sales %>% rename(Pizza_ID = pizza_id, Order_ID = order_id, Pizza_Nam
e_ID = pizza_name_id, Quantity = quantity, Order_Date = order_date,
      Order_Time = order_time, Unit_Price = unit_price, Total_Sale = total_price, Pizza
_Size = pizza_size, Pizza_Category = pizza_category,
      Pizza_Ingredients = pizza_ingredients, Pizza_Name = pizza_name)
```

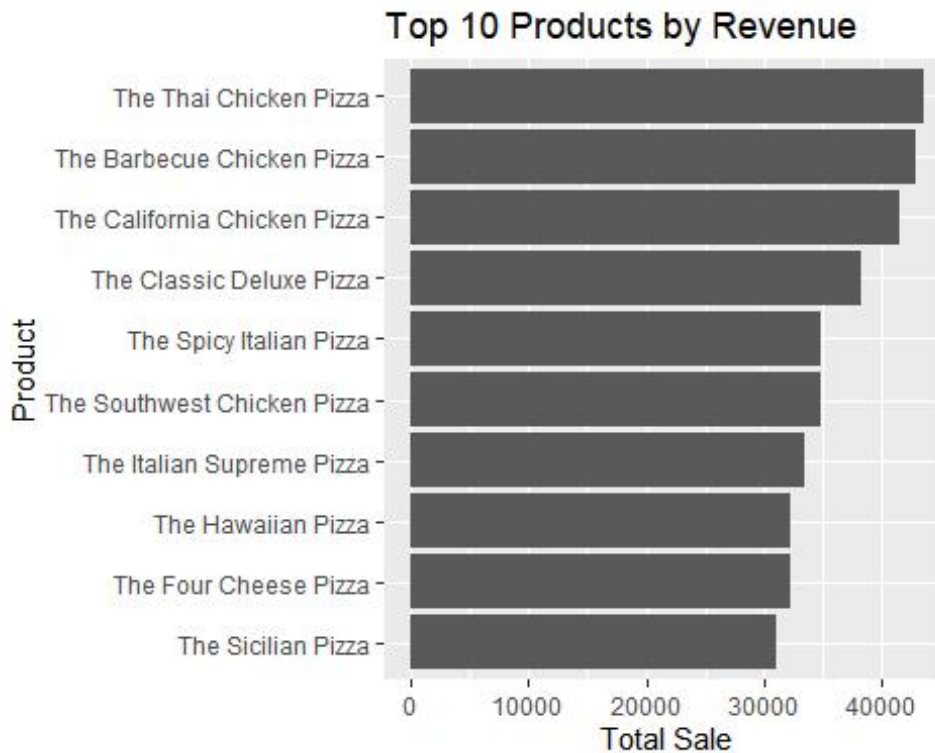
##identifying the top products based on total revenue

```
Pizza_sales %>%  
  group_by(Pizza_Name) %>%  
  summarise(Total_Sale = sum(Total_Sale, na.rm = TRUE)) %>%  
  arrange(desc(Total_Sale)) %>%  
  head(10)
```

```
## # A tibble: 10 × 2  
##   Pizza_Name      Total_Sale  
##   <chr>          <dbl>  
## 1 The Thai Chicken Pizza    43434.  
## 2 The Barbecue Chicken Pizza 42768  
## 3 The California Chicken Pizza 41410.  
## 4 The Classic Deluxe Pizza   38180.  
## 5 The Spicy Italian Pizza    34831.  
## 6 The Southwest Chicken Pizza 34706.  
## 7 The Italian Supreme Pizza  33477.  
## 8 The Hawaiian Pizza        32273.  
## 9 The Four Cheese Pizza      32266.  
## 10 The Sicilian Pizza        30940.
```

##visualizing the top 10 products sold

```
top_products <- Pizza_sales %>%  
  group_by(Pizza_Name) %>%  
  summarise(Total_Sale = sum(Total_Sale)) %>%  
  arrange(desc(Total_Sale)) %>%  
  head(10)  
ggplot(top_products, aes(x = reorder(Pizza_Name, Total_Sale), y = Total_Sale)) +  
  geom_col() +  
  coord_flip() +  
  labs(title = "Top 10 Products by Revenue",  
        x = "Product",  
        y = "Total Sale")
```



##Identifying the top products based on quantity sold

```
Pizza_sales %>%
  group_by(Pizza_Name) %>%
  summarise(Quantity = sum(Quantity, na.rm = TRUE)) %>%
  arrange(desc(Quantity)) %>%
  head(10)
```

```
## # A tibble: 10 × 2
##   Pizza_Name      Quantity
##   <chr>          <int>
## 1 The Classic Deluxe Pizza    2453
## 2 The Barbecue Chicken Pizza  2432
## 3 The Hawaiian Pizza         2422
## 4 The Pepperoni Pizza        2418
## 5 The Thai Chicken Pizza     2371
## 6 The California Chicken Pizza 2370
## 7 The Sicilian Pizza         1938
## 8 The Spicy Italian Pizza     1924
## 9 The Southwest Chicken Pizza 1917
## 10 The Big Meat Pizza        1914
```

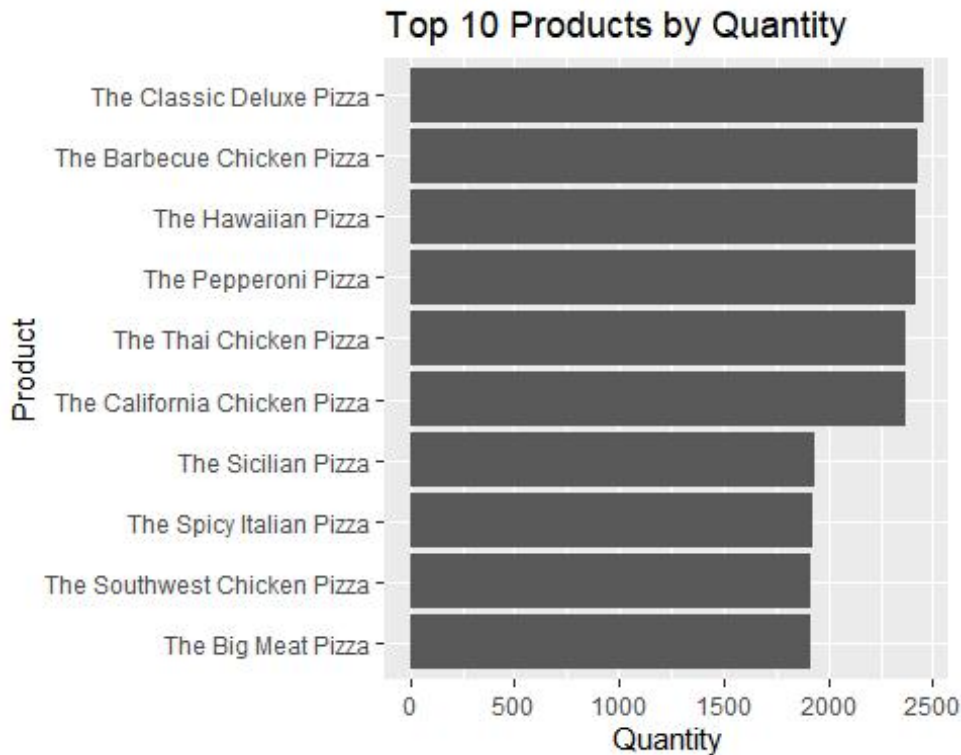
##Visualizing the top products by quantity

```
top_quantity <- Pizza_sales %>%
  group_by(Pizza_Name) %>%
  summarise(Quantity = sum(Quantity, na.rm = TRUE)) %>%
  arrange(desc(Quantity)) %>%
```

```

head(10)
ggplot(top_quantity, aes(x = reorder(Pizza_Name, Quantity), y = Quantity)) +
  geom_col() +
  coord_flip() +
  labs(title = "Top 10 Products by Quantity",
       x = "Product",
       y = "Quantity")

```



##Identifying the top 1 product sold

```

Pizza_sales %>%
  group_by(Pizza_Name) %>%
  summarise(Total_Sale = sum(Total_Sale)) %>%
  slice_max(Total_Sale, n = 1)

```

```

## # A tibble: 1 × 2
##   Pizza_Name      Total_Sale
##   <chr>          <dbl>
## 1 The Thai Chicken Pizza  43434.

```

##Calculating the average order value

```

order_totals <- Pizza_sales %>%
  mutate(Order_ID = Quantity * Unit_Price) %>%
  group_by(Order_ID) %>%
  summarise(TotalOrderValue = sum(Order_ID, na.rm = TRUE)) ##creating order total
mean(order_totals$TotalOrderValue)

```

[1] 14604.64