

# Análisis 2017

STDT Ing. George Albadr

2024-11-04

```
rm(list = ls())

knitr::opts_chunk$set(echo = TRUE, warning = FALSE, message = FALSE)
# Librerías
library(readr)
library(lubridate)
library(dplyr)
library(ggplot2)
library(tidyr)
library(scales)
library(knitr)
library(geosphere)
library(kableExtra)

# Cargar los datos
data <- read_csv("c1.csv")

# Convertir la columna 'Fecha' al formato de fecha correcto
data$Fecha <- as.Date(data$Fecha, format = "%d-%m-%y")

# Limpieza de Datos en el mismo orden

# Lista de columnas monetarias y sus nuevos nombres
monetary_columns <- c("Camion_5", "Pickup", "Moto", "factura",
                      "directoCamion_5", "directoPickup", "directoMoto",
                      "fijoCamion_5", "fijoPickup", "fijoMoto")
new_names <- c("Camion", "Pickup", "Moto", "factura",
              "directoCamion", "directoPickup", "directoMoto",
              "fijoCamion", "fijoPickup", "fijoMoto")

# Limpieza y conversión de columnas monetarias
for (i in seq_along(monetary_columns)) {
  col <- monetary_columns[i]
  new_col <- new_names[i]
  data[[col]] <- gsub("Q-", "", data[[col]])
  data[[col]] <- gsub("Q", "", data[[col]])
  data[[col]][data[[col]] == ""] <- NA
  data[[col]] <- as.numeric(data[[col]])
  colnames(data)[colnames(data) == col] <- new_col
}

# Conversión de 'x' a Booleanos para las columnas de tiempo
```

```

time_columns <- c("5-30", "30-45", "45-75", "75-120", "120+")

for (col in time_columns) {
  data[[col]] <- ifelse(data[[col]] == "x", TRUE, FALSE)
}

# Eliminar columnas que no sirven de nada
data <- data[, !names(data) %in% c("...23", "...24", "...25", "...26", "...27", "...28")]

# Verificar la estructura de los datos
str(data)

## tibble [263,725 x 22] (S3: tbl_df/tbl/data.frame)
## $ Fecha      : Date[1:263725], format: "2017-12-10" "2017-03-19" ...
## $ ID         : num [1:263725] 368224 368224 368224 368224 748633 ...
## $ Camion     : num [1:263725] NA NA NA NA NA NA NA NA NA NA ...
## $ Pickup     : num [1:263725] 271 262 230 269 233 ...
## $ Moto       : num [1:263725] NA NA NA NA NA NA NA NA NA NA ...
## $ Cod        : chr [1:263725] "VERIFICACION_MEDIDORES" "REVISION_TRANSFORMADOR" "REVISION" "REVIS
## $ origen     : num [1:263725] 150277 150277 150277 150224 150277 ...
## $ Lat        : num [1:263725] 15.5 15.5 15.5 15.5 14.7 ...
## $ Long       : num [1:263725] -89.7 -89.7 -89.7 -89.7 -90.9 ...
## $ factura    : num [1:263725] 317 267 236 290 248 ...
## $ directoCamion: num [1:263725] NA NA NA NA NA NA NA NA NA NA ...
## $ directoPickup: num [1:263725] 168 173 136 175 154 ...
## $ directoMoto : num [1:263725] NA NA NA NA NA NA NA NA NA NA ...
## $ fijoCamion  : num [1:263725] NA NA NA NA NA NA NA NA NA NA ...
## $ fijoPickup  : num [1:263725] 102.9 89.2 94.3 94.1 79.3 ...
## $ fijoMoto    : num [1:263725] NA NA NA NA NA NA NA NA NA NA ...
## $ height     : num [1:263725] 8 8 8 8 8 12 12 12 10 8 ...
## $ 5-30       : logi [1:263725] NA NA NA NA NA NA NA ...
## $ 30-45      : logi [1:263725] NA NA NA NA NA NA NA ...
## $ 45-75      : logi [1:263725] NA NA NA NA NA NA NA ...
## $ 75-120     : logi [1:263725] NA NA NA NA NA NA NA ...
## $ 120+       : logi [1:263725] TRUE TRUE TRUE TRUE TRUE TRUE ...

## Filtrar datos del 2017
data_2017 <- data %>% filter(year(Fecha) == 2017)

# Calcular costos directos y fijos
data_2017 <- data_2017 %>%
  mutate(
    Costo_Directo = rowSums(select(., directoCamion, directoPickup, directoMoto), na.rm = TRUE),
    Costo_Fijo = rowSums(select(., fijoCamion, fijoPickup, fijoMoto), na.rm = TRUE),
    Costo_Ventas = Costo_Directo + Costo_Fijo,
    Margen_Bruto = factura - Costo_Ventas
  )

# Agregar columna del mes
data_2017$Mes <- month(data_2017$Fecha, label = TRUE, abbr = TRUE)

# Calcular ventas totales por mes
ventas_por_mes <- data_2017 %>%
  group_by(Mes) %>%

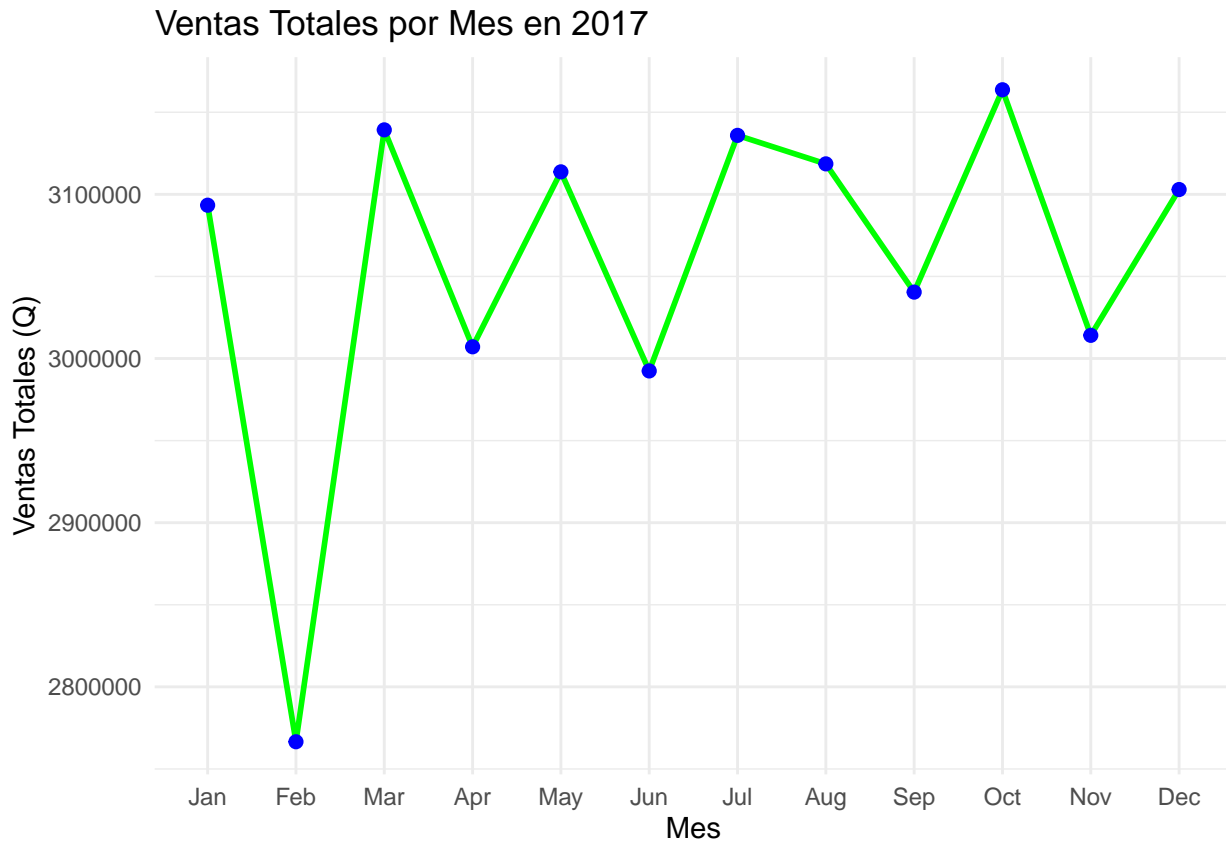
```

```

summarise(Ventas_Totales = sum(factura, na.rm = TRUE))

# Generar un gráfico de líneas de ventas totales por mes
ggplot(ventas_por_mes, aes(x = Mes, y = Ventas_Totales, group = 1)) +
  geom_line(color = "green", size = 1) +
  geom_point(color = "blue", size = 2) +
  labs(title = "Ventas Totales por Mes en 2017",
       x = "Mes",
       y = "Ventas Totales (Q)") +
  theme_minimal()

```



```

# Cargar el paquete scales para formatear números
library(scales)

# Calcular ingresos por tipo de unidad
ingresos_por_unidad <- data_2017 %>%
  summarise(
    Ingreso_Camion = sum(Camion, na.rm = TRUE),
    Ingreso_Pickup = sum(Pickup, na.rm = TRUE),
    Ingreso_Moto = sum(Moto, na.rm = TRUE)
  )

# Formatear los ingresos con comas
ingresos_por_unidad_formateado <- ingresos_por_unidad %>%
  mutate(
    Ingreso_Camion = comma(Ingreso_Camion, big.mark = ","),
    Ingreso_Pickup = comma(Ingreso_Pickup, big.mark = ","),

```

```

    Ingreso_Moto = comma(Ingreso_Moto, big.mark = ",")
  )

# Mostrar ingresos por unidad
print("Ingresos por Tipo de Unidad en 2017:")

## [1] "Ingresos por Tipo de Unidad en 2017:"
print(ingresos_por_unidad_formateado)

## # A tibble: 1 x 3
##   Ingreso_Camion Ingreso_Pickup Ingreso_Moto
##   <chr>          <chr>          <chr>
## 1 8,658,363      19,121,908      393,748

# Calcular tarifas promedio por unidad
tarifa_promedio_camion <- mean(data_2017$Camion, na.rm = TRUE)
tarifa_promedio_pickup <- mean(data_2017$Pickup, na.rm = TRUE)
tarifa_promedio_moto <- mean(data_2017$Moto, na.rm = TRUE)

# Mostrar tarifas promedio formateadas con comas
print(paste("Tarifa Promedio Camión: Q", comma(round(tarifa_promedio_camion, 2))))

## [1] "Tarifa Promedio Camión: Q 139"
print(paste("Tarifa Promedio Pickup: Q", comma(round(tarifa_promedio_pickup, 2))))

## [1] "Tarifa Promedio Pickup: Q 98"
print(paste("Tarifa Promedio Moto: Q", comma(round(tarifa_promedio_moto, 2))))

## [1] "Tarifa Promedio Moto: Q 69"

# Calcular el margen operativo total
margen_operativo_total <- sum(data_2017$Margen_Bruto, na.rm = TRUE)

# Verificar si estamos en números rojos
if (margen_operativo_total < 0) {
  print("La empresa está en números rojos en 2017.")
} else {
  print("La empresa tiene un margen operativo positivo en 2017.")
}

## [1] "La empresa tiene un margen operativo positivo en 2017."
print(paste("Margen Operativo Total en 2017: Q", format(round(margen_operativo_total, 2), big.mark = ",")

## [1] "Margen Operativo Total en 2017: Q 8,514,077"

# Agrupar la facturación total por ID y ordenar de mayor a menor
facturacion_por_id <- data_2017 %>%
  group_by(ID) %>%
  summarise(Facturacion_Total = sum(factura, na.rm = TRUE)) %>%
  arrange(desc(Facturacion_Total)) %>%
  ungroup()

# Seleccionar los top 10 IDs con mayor facturación
top_10_ids <- facturacion_por_id %>% slice_head(n = 10)

```

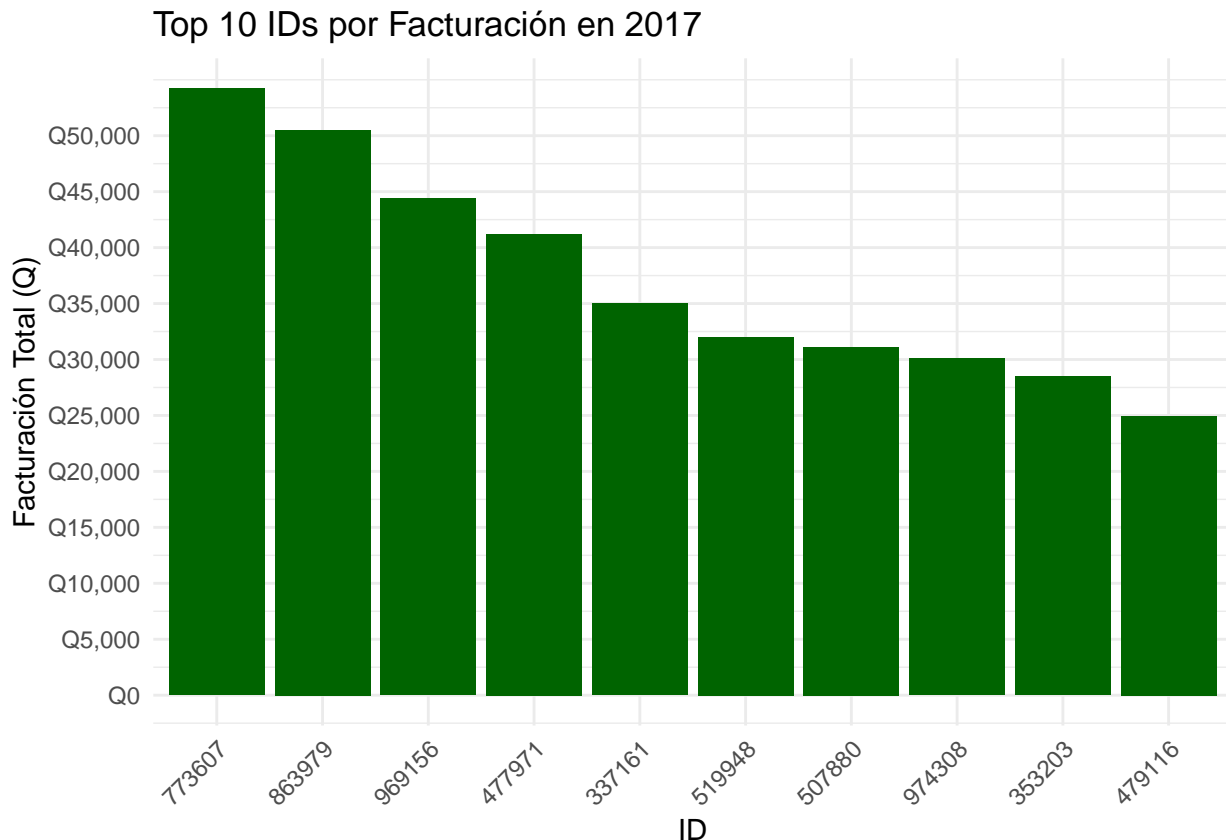
```

# Convertir IDs a factor para mantener el orden en el gráfico
top_10_ids$ID <- factor(top_10_ids$ID, levels = top_10_ids$ID)

# Cargar la librería scales para formatear los números
library(scales)

# Generar un gráfico de barras de los top 10 IDs por facturación con ejes ajustados
ggplot(top_10_ids, aes(x = ID, y = Facturacion_Total)) +
  geom_bar(stat = "identity", fill = "darkgreen") +
  labs(title = "Top 10 IDs por Facturación en 2017",
       x = "ID",
       y = "Facturación Total (Q)") +
  scale_y_continuous(
    labels = function(x) { paste0("Q", comma_format(big.mark = ",")(x)) },
    breaks = seq(0, max(top_10_ids$Facturacion_Total), by = 5000)
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

```



```

# Calcular costos de mantenimiento por ID y ordenar de mayor a menor
mantenimiento_por_id <- data_2017 %>%
  group_by(ID) %>%
  summarise(Costo_Mantenimiento = sum(Costo_Ventas, na.rm = TRUE)) %>%
  arrange(desc(Costo_Mantenimiento)) %>%
  ungroup()

# Calcular el total de costos de mantenimiento

```

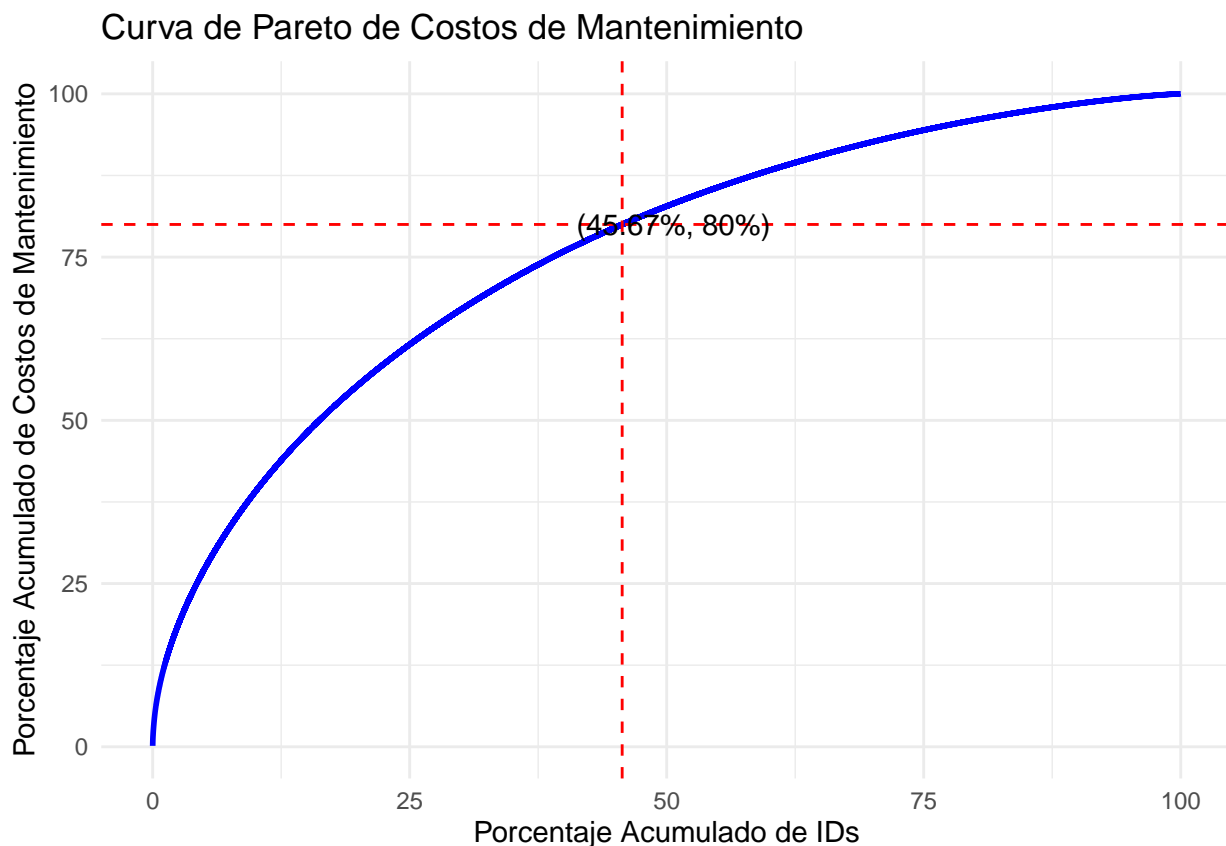
```
total_costo_mantenimiento <- sum(mantenimiento_por_id$Costo_Mantenimiento)

# Calcular porcentaje acumulado de costos y porcentaje acumulado de IDs
mantenimiento_por_id <- mantenimiento_por_id %>%
  mutate(
    Porcentaje_Costo_Acum = cumsum(Costo_Mantenimiento) / total_costo_mantenimiento * 100,
    Porcentaje_ID_Acum = (row_number()) / n() * 100
  )

# Encontrar el punto donde se alcanza el 80% del costo acumulado
punto_80 <- mantenimiento_por_id %>% filter(Porcentaje_Costo_Acum >= 80) %>% slice_head(n = 1)

# Calcular el porcentaje de IDs que representan el 80% de los costos
porcentaje_ids_80 <- punto_80$Porcentaje_ID_Acum

# Gráfico de la curva de Pareto
ggplot(mantenimiento_por_id, aes(x = Porcentaje_ID_Acum, y = Porcentaje_Costo_Acum)) +
  geom_line(color = "blue", size = 1) +
  geom_vline(xintercept = porcentaje_ids_80, linetype = "dashed", color = "red") +
  geom_hline(yintercept = 80, linetype = "dashed", color = "red") +
  annotate("text", x = porcentaje_ids_80 + 5, y = 80,
    label = paste0("(", round(porcentaje_ids_80, 2), "%, 80%)", color = "black") +
  labs(title = "Curva de Pareto de Costos de Mantenimiento",
    x = "Porcentaje Acumulado de IDs",
    y = "Porcentaje Acumulado de Costos de Mantenimiento") +
  theme_minimal()
```



```
# Asignar un tiempo promedio a cada rango
```

```
data_2017 <- data_2017 %>%
```

```
  mutate(
    Tiempo_Estimado = case_when(
      `5-30` == TRUE ~ 17.5,
      `30-45` == TRUE ~ 37.5,
      `45-75` == TRUE ~ 60,
      `75-120` == TRUE ~ 97.5,
      `120+` == TRUE ~ 150,
      TRUE ~ NA_real_
    )
  )
```

```
# Eliminar filas con NA en Tiempo_Estimado
```

```
data_2017_clean <- data_2017 %>%
```

```
  filter(!is.na(Tiempo_Estimado))
```

```
# Calcular estadísticas descriptivas
```

```
descriptivas <- data_2017_clean %>%
```

```
  summarise(
    Media = mean(Tiempo_Estimado, na.rm = TRUE),
    Mediana = median(Tiempo_Estimado, na.rm = TRUE),
    Moda = as.numeric(names(sort(table(Tiempo_Estimado), decreasing = TRUE)[1])),
    Desviacion_Estandar = sd(Tiempo_Estimado, na.rm = TRUE),
    Minimo = min(Tiempo_Estimado, na.rm = TRUE),
    Maximo = max(Tiempo_Estimado, na.rm = TRUE)
  )
```

```
print("Estadísticas Descriptivas de Tiempo Estimado:")
```

```
## [1] "Estadísticas Descriptivas de Tiempo Estimado:"
```

```
print(descriptivas)
```

```
## # A tibble: 1 x 6
```

```
##   Media Mediana Moda Desviacion_Estandar Minimo Maximo
```

```
##   <dbl>   <dbl> <dbl>           <dbl>   <dbl>   <dbl>
```

```
## 1  73.9    97.5  97.5             39.8    17.5    150
```

```
# Calcular la frecuencia total para porcentajes
```

```
total_frecuencia <- sum(table(data_2017_clean$Tiempo_Estimado))
```

```
# Gráfico de distribución de tiempos estimados con porcentajes
```

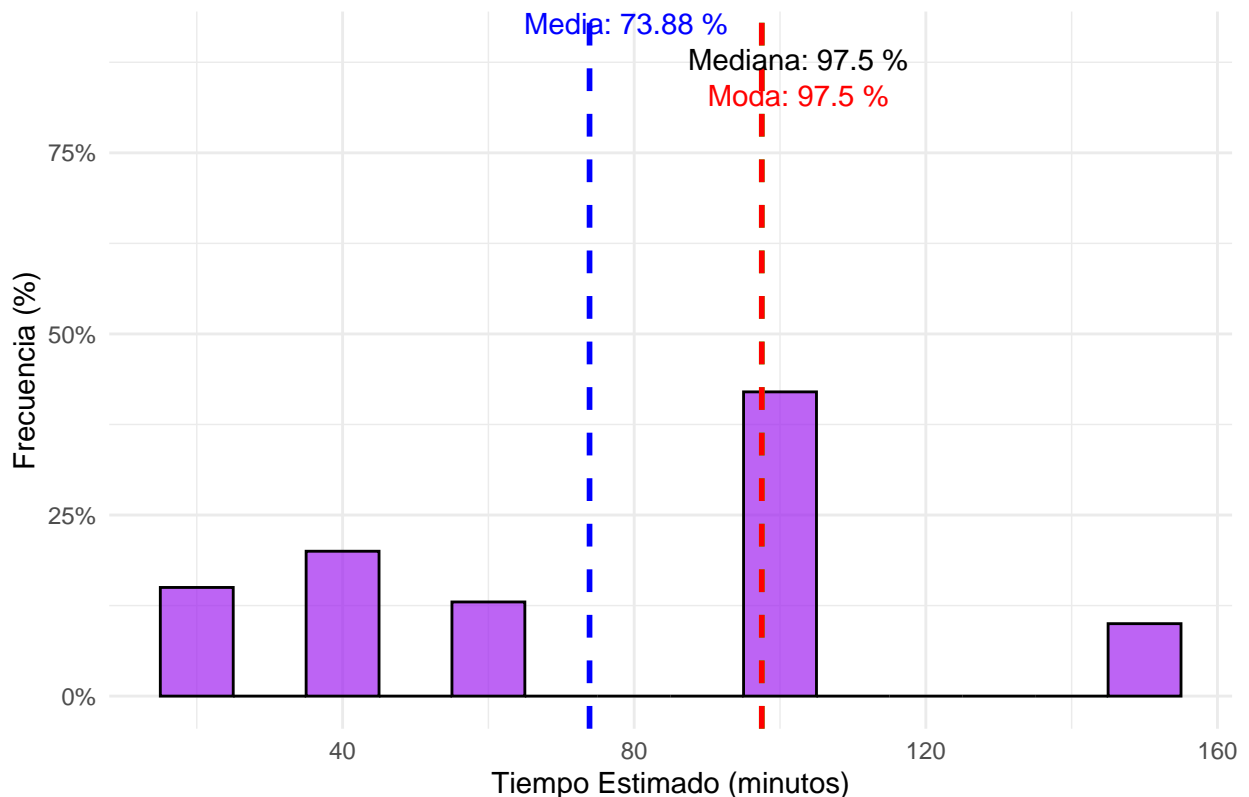
```
ggplot(data_2017_clean, aes(x = Tiempo_Estimado)) +
  geom_histogram(aes(y = ..count../sum(..count..) * 100), binwidth = 10, fill = "purple", color = "black") +
  geom_vline(aes(xintercept = Media), data = descriptivas, color = "blue", linetype = "dashed", size = 1) +
  geom_vline(aes(xintercept = Mediana), data = descriptivas, color = "green", linetype = "dashed", size = 1) +
  geom_vline(aes(xintercept = Moda), data = descriptivas, color = "red", linetype = "dashed", size = 1) +
  scale_y_continuous(labels = percent_format(scale = 1)) +
  labs(title = "Distribución de Tiempos Estimados de Recorrido en 2017",
    x = "Tiempo Estimado (minutos)",
    y = "Frecuencia (%)") +
  theme_minimal() +
  annotate("text", x = descriptivas$Media + 5, y = 90,
    label = paste("Media:", round(descriptivas$Media, 2), "%"), color = "blue", angle = 0, vjust = "bottom")
```

```

annotate("text", x = descriptivas$Mediana + 5, y = 85,
        label = paste("Mediana:", round(descriptivas$Mediana, 2), "%"), color = "black", angle = 0, vjust = 1),
annotate("text", x = descriptivas$Moda + 5, y = 80,
        label = paste("Moda:", round(descriptivas$Moda, 2), "%"), color = "red", angle = 0, vjust = 1)

```

## Distribución de Tiempos Estimados de Recorrido en 2017



### ##Paso 1: Definir los Centros de Distribución

Primero, debes definir los cuatro centros de distribución con sus respectivas coordenadas (latitud y longitud). Asegúrate de reemplazar las coordenadas de ejemplo con las reales de tus centros.

```

# Definir los centros de distribución con coordenadas reales utilizando k-means clustering
# Seleccionar las columnas de latitud y longitud, asegurando que sean numéricas y sin NA
coords <- data_2017 %>%
  select(Lat, Long) %>%
  filter(!is.na(Lat) & !is.na(Long))

# Verificar el número de centros a crear
k <- 4 # Número de centros de distribución

# Realizar clustering con k-means para identificar 4 centros
set.seed(123) # Para reproducibilidad
kmeans_result <- kmeans(coords, centers = k, nstart = 25)

# Crear dataframe de centros de distribución basados en los centros del clustering
centros_distribucion <- data.frame(
  Centro = paste0("Centro", 1:k),
  Lat = round(kmeans_result$centers[, "Lat"], 6),
  Long = round(kmeans_result$centers[, "Long"], 6)
)

```



```
)
```

```
# Mostrar los centros de distribución
```

```
kable(centros_distribucion, caption = "Centros de Distribución Identificados con sus Coordenadas Reales",  
      kable_styling(full_width = FALSE, position = "center"))
```

Table 1: Centros de Distribución Identificados con sus Coordenadas Reales

| Centro  | Lat      | Long      |
|---------|----------|-----------|
| Centro1 | 15.50703 | -90.49530 |
| Centro2 | 14.49850 | -90.50241 |
| Centro3 | 14.49200 | -89.50516 |
| Centro4 | 15.48897 | -89.49332 |

##Paso 2: Asignar Servicios al Centro de Distribución Más Cercano

Utilizaremos la función `distHaversine` del paquete `geosphere` para calcular la distancia entre cada servicio y los centros de distribución, asignando cada servicio al centro más cercano.

```
# Asegurarse de que las columnas Lat y Long son numéricas
```

```
data_2017 <- data_2017 %>%
```

```
  mutate(  
    Lat = as.numeric(Lat),  
    Long = as.numeric(Long)  
  )
```

```
# Función para asignar cada servicio al centro más cercano
```

```
asignar_centro <- function(lat, long, centros) {  
  # Crear una matriz de coordenadas para el servicio  
  servicio_coord <- c(long, lat)  
  
  # Crear una matriz de coordenadas para los centros  
  centros_coords <- as.matrix(centros[, c("Long", "Lat")])  
  
  # Calcular distancias a todos los centros (en metros)  
  distancias <- distHaversine(servicio_coord, centros_coords)  
  
  # Retornar el nombre del centro más cercano  
  centros$Centro[which.min(distancias)]  
}
```

```
# Asignar centro a cada servicio
```

```
data_2017 <- data_2017 %>%  
  rowwise() %>%  
  mutate(Centro_Distribucion = asignar_centro(Lat, Long, centros_distribucion)) %>%  
  ungroup()
```

```
# Verificar la asignación de servicios a cada centro
```

```
frecuencia_centros <- data_2017 %>%  
  group_by(Centro_Distribucion) %>%  
  summarise(  
    Numero_Servicios = n(),
```

```

  Porcentaje_Servicios = round((n() / nrow(data_2017)) * 100, 2)
)

# Formatear las cifras
frecuencia_centros <- frecuencia_centros %>%
  mutate(
    Numero_Servicios = comma(Numero_Servicios, big.mark = ","),
    Porcentaje_Servicios = paste0(Porcentaje_Servicios, "%")
  )

# Mostrar la tabla de asignación
kable(frecuencia_centros,
      caption = "Frecuencia de Servicios por Centro de Distribución") %>%
  kable_styling(full_width = FALSE, position = "center")

```

Table 2: Frecuencia de Servicios por Centro de Distribución

| Centro_Distribucion | Numero_Servicios | Porcentaje_Servicios |
|---------------------|------------------|----------------------|
| Centro1             | 65,031           | 24.66%               |
| Centro2             | 66,470           | 25.2%                |
| Centro3             | 65,561           | 24.86%               |
| Centro4             | 66,663           | 25.28%               |

### ### Paso 3: Resumir el Margen Bruto por Centro de Distribución

Generaremos una tabla que resume el Total de Margen Bruto, el Número de Servicios y el Margen Promedio por cada centro de distribución.

```

# Calcular total y margen promedio por centro
margen_por_centro <- data_2017 %>%
  group_by(Centro_Distribucion) %>%
  summarise(
    Total_Margen_Bruto = sum(Margen_Bruto, na.rm = TRUE),
    Servicios = n(),
    Margen_Promedio = mean(Margen_Bruto, na.rm = TRUE)
  ) %>%
  mutate(
    Total_Margen_Bruto = comma(Total_Margen_Bruto, big.mark = ","),
    Margen_Promedio = comma(round(Margen_Promedio, 2), big.mark = ",")
  )

# Mostrar la tabla formateada
kable(margen_por_centro,
      caption = "Resumen del Margen Bruto por Centro de Distribución") %>%
  kable_styling(full_width = FALSE, position = "center")

```

Table 3: Resumen del Margen Bruto por Centro de Distribución

| Centro_Distribucion | Total_Margen_Bruto | Servicios | Margen_Promedio |
|---------------------|--------------------|-----------|-----------------|
| Centro1             | 2,097,066          | 65031     | 32.25           |
| Centro2             | 2,148,871          | 66470     | 32.33           |
| Centro3             | 2,126,025          | 65561     | 32.43           |

##verificacion

```
# Inspeccionar una muestra de asignaciones
muestra_asignacion <- data_2017 %>%
  select(ID, Lat, Long, Centro_Distribucion) %>%
  slice_head(n = 20)

# Mostrar la muestra
kable(muestra_asignacion,
      caption = "Muestra de Asignación de Servicios a Centros de Distribución") %>%
kable_styling(full_width = FALSE, position = "center")
```

Table 4: Muestra de Asignación de Servicios a Centros de Distribución

| ID     | Lat      | Long      | Centro_Distribucion |
|--------|----------|-----------|---------------------|
| 368224 | 15.46333 | -89.72565 | Centro4             |
| 368224 | 15.46333 | -89.72565 | Centro4             |
| 368224 | 15.46333 | -89.72565 | Centro4             |
| 368224 | 15.46333 | -89.72565 | Centro4             |
| 748633 | 14.72568 | -90.89644 | Centro2             |
| 599434 | 15.93645 | -89.04138 | Centro4             |
| 599434 | 15.93645 | -89.04138 | Centro4             |
| 599434 | 15.93645 | -89.04138 | Centro4             |
| 790749 | 15.95877 | -90.89046 | Centro1             |
| 760915 | 14.39340 | -90.54944 | Centro2             |
| 760915 | 14.39340 | -90.54944 | Centro2             |
| 576258 | 15.20378 | -90.81217 | Centro1             |
| 814417 | 14.43319 | -90.38244 | Centro2             |
| 449894 | 15.62738 | -90.16696 | Centro1             |
| 565298 | 14.46072 | -90.79400 | Centro2             |
| 460650 | 14.07410 | -89.95981 | Centro3             |
| 730040 | 14.95062 | -89.89759 | Centro3             |
| 730040 | 14.95062 | -89.89759 | Centro3             |
| 730040 | 14.95062 | -89.89759 | Centro3             |
| 730040 | 14.95062 | -89.89759 | Centro3             |

```
# Recalcular y formatear el resumen del margen bruto por centro
margen_por_centro <- data_2017 %>%
  group_by(Centro_Distribucion) %>%
  summarise(
    Total_Margen_Bruto = sum(Margen_Bruto, na.rm = TRUE),
    Servicios = n(),
    Margen_Promedio = mean(Margen_Bruto, na.rm = TRUE)
  ) %>%
  mutate(
    Total_Margen_Bruto = comma(Total_Margen_Bruto, big.mark = ","),
    Margen_Promedio = comma(round(Margen_Promedio, 2), big.mark = ",")
  )
```

```
# Mostrar la tabla formateada
kable(margen_por_centro,
      caption = "Resumen del Margen Bruto por Centro de Distribución") %>%
kable_styling(full_width = FALSE, position = "center")
```

Table 5: Resumen del Margen Bruto por Centro de Distribución

| Centro_Distribucion | Total_Margen_Bruto | Servicios | Margen_Promedio |
|---------------------|--------------------|-----------|-----------------|
| Centro1             | 2,097,066          | 65031     | 32.25           |
| Centro2             | 2,148,871          | 66470     | 32.33           |
| Centro3             | 2,126,025          | 65561     | 32.43           |
| Centro4             | 2,142,116          | 66663     | 32.13           |

```
##Tabla de estrategias
```

```
# Crear una tabla de estrategias
estrategias <- data.frame(
  Estrategia = c(
    "Optimizar Centros con Alto Margen Bruto",
    "Reducir Costos en Centros con Bajo Margen Bruto",
    "Redistribuir Servicios",
    "Implementar Programas de Fidelización",
    "Monitorear y Evaluar Continuamente"
  ),
  Accion = c(
    "Analizar y replicar mejores prácticas en todos los centros.",
    "Identificar y eliminar costos innecesarios.",
    "Equilibrar la carga de trabajo entre centros.",
    "Fomentar la lealtad de clientes en centros clave.",
    "Establecer KPIs y realizar seguimiento regular."
  ),
  Impacto_Esperado = c(
    "Aumento de la eficiencia y márgenes en todos los centros.",
    "Incremento en los márgenes brutos mediante la reducción de costos.",
    "Mejor utilización de recursos y mayor eficiencia operativa.",
    "Incremento en ventas y márgenes brutos.",
    "Capacidad de mantener y mejorar márgenes brutos."
  )
)

# Mostrar la tabla de estrategias
kable(estrategias,
      caption = "Estrategias para Aumentar el Margen Bruto por Centro de Distribución") %>%
kable_styling(full_width = FALSE, position = "center")
```

Table 6: Estrategias para Aumentar el Margen Bruto por Centro de Distribución

| Estrategia                                      | Accion  | Impacto    |
|---|---|------------|
| Optimizar Centros con Alto Margen Bruto         | Analizar y replicar mejores prácticas en todos los centros. | Aumento    |
| Reducir Costos en Centros con Bajo Margen Bruto | Identificar y eliminar costos innecesarios.                 | Incremento |
| Redistribuir Servicios                          | Equilibrar la carga de trabajo entre centros.               | Mejor u    |

```
# Identificar servicios con margen bruto negativo
servicios_perdida <- data_2017 %>% filter(Margen_Bruto < 0)

# Número y porcentaje de servicios con pérdidas
num_servicios_perdida <- nrow(servicios_perdida)
total_servicios <- nrow(data_2017)
porcentaje_perdida <- (num_servicios_perdida / total_servicios) * 100

print(paste("Número de servicios con margen bruto negativo:", num_servicios_perdida))

## [1] "Número de servicios con margen bruto negativo: 0"

print(paste("Porcentaje de servicios con pérdidas:", round(porcentaje_perdida, 2), "%"))

## [1] "Porcentaje de servicios con pérdidas: 0 %"
```

En esta parte realizamos un analisis de la acoplacion de un programa de fidelizacion con el objetivo de aumentar las ventas actuales netas en un 10% brindando un 2% de descuento en el total de ventas para incentivar aun mas a los clientes.

```
# Actualizamos los datos de los top 10 IDs
top_10_ids <- top_10_ids %>%
  mutate(
    Ventas_Actuales = Facturacion_Total,
    Incremento_Ventas = Ventas_Actuales * 0.10,
    Ventas_Nuevas = Ventas_Actuales + Incremento_Ventas,
    Descuento = Ventas_Nuevas * 0.02,
    Facturacion_Despues_Descuento = Ventas_Nuevas - Descuento,
    Costo_Ventas_Actual = Ventas_Actuales * 0.70,
    Costo_Ventas_Nuevo = Ventas_Nuevas * 0.70,
    Margen_Bruto_Actual = Ventas_Actuales - Costo_Ventas_Actual,
    Margen_Bruto_Nuevo = Facturacion_Despues_Descuento - Costo_Ventas_Nuevo,
    Diferencia_Margen_Bruto = Margen_Bruto_Nuevo - Margen_Bruto_Actual
  )

# Formateamos las columnas numéricas con comas
top_10_ids_formateado <- top_10_ids %>%
  mutate_at(vars(Ventas_Actuales, Ventas_Nuevas, Margen_Bruto_Actual, Margen_Bruto_Nuevo, Diferencia_Margen_Bruto), fun = function(x) format(x, big.mark = ","))

# Mostrar los resultados
top_10_ids_formateado %>%
  select(ID, Ventas_Actuales, Ventas_Nuevas, Margen_Bruto_Actual, Margen_Bruto_Nuevo, Diferencia_Margen_Bruto)

## # A tibble: 10 x 6
##   ID      Ventas_Actuales Ventas_Nuevas Margen_Bruto_Actual Margen_Bruto_Nuevo
##   <fct>   <chr>           <chr>           <chr>           <chr>
## 1 773607 54,202             59,623          16,261          16,694
## 2 863979 50,494             55,544          15,148          15,552
```

```
## 3 969156 44,404      48,845      13,321      13,677
## 4 477971 41,224      45,346      12,367      12,697
## 5 337161 34,991      38,490      10,497      10,777
## 6 519948 32,008      35,209      9,602       9,858
## 7 507880 31,123      34,235      9,337       9,586
## 8 974308 30,087      33,096      9,026       9,267
## 9 353203 28,458      31,303      8,537       8,765
## 10 479116 24,938      27,432      7,481       7,681
## # i 1 more variable: Diferencia_Margen_Bruto <chr>
```

Aqui se habla de como un imacto en una economia a escala y reduccion del 5% del costo de ventas al margen bruto en general

```
#Supongamos que el costo de ventas disminuye al 65% debido a economias de escala
```

```
# Actualizamos el costo de ventas nuevo
```

```
top_10_ids <- top_10_ids %>%
  mutate(
    Costo_Ventas_Nuevo = Ventas_Nuevas * 0.65
  )
```

```
# Recalculamos el Margen Bruto Nuevo y la Diferencia del Margen Bruto
```

```
top_10_ids <- top_10_ids %>%
  mutate(
    Margen_Bruto_Nuevo = Facturacion_Despues_Descuento - Costo_Ventas_Nuevo,
    Diferencia_Margen_Bruto = Margen_Bruto_Nuevo - Margen_Bruto_Actual
  )
```

```
# Formateamos las columnas numéricas con comas
```

```
top_10_ids_formateado <- top_10_ids %>%
  mutate_at(vars(Ventas_Actuales, Ventas_Nuevas, Margen_Bruto_Actual, Margen_Bruto_Nuevo, Diferencia_Ma
```

```
# Mostrar los resultados actualizados
```

```
top_10_ids_formateado %>%
  select(ID, Ventas_Actuales, Ventas_Nuevas, Margen_Bruto_Actual, Margen_Bruto_Nuevo, Diferencia_Margen
```

```
## # A tibble: 10 x 6
```

```
##   ID      Ventas_Actuales Ventas_Nuevas Margen_Bruto_Actual Margen_Bruto_Nuevo
##   <fct> <chr>              <chr>          <chr>              <chr>
## 1 773607 54,202              59,623          16,261             19,675
## 2 863979 50,494              55,544          15,148             18,329
## 3 969156 44,404              48,845          13,321             16,119
## 4 477971 41,224              45,346          12,367             14,964
## 5 337161 34,991              38,490          10,497             12,702
## 6 519948 32,008              35,209          9,602              11,619
## 7 507880 31,123              34,235          9,337              11,298
## 8 974308 30,087              33,096          9,026              10,922
## 9 353203 28,458              31,303          8,537              10,330
## 10 479116 24,938              27,432          7,481              9,052
## # i 1 more variable: Diferencia_Margen_Bruto <chr>
```

Aquí se hablaron posibles escenarios en el incremento de y descuentos de ventas en margen bruto para los clientes para ver más opciones para aumentar el margen bruto.

```
# Definir la función probar_escenario sin formatear
probar_escenario <- function(incremento_ventas_pct, descuento_pct, costo_ventas_pct_actual, costo_ventas_pct_nuevo, top_10_ids) {
  top_10_ids %>%
    mutate(
      Ventas_Actuales = Facturacion_Total,
      Incremento_Ventas = Ventas_Actuales * incremento_ventas_pct,
      Ventas_Nuevas = Ventas_Actuales + Incremento_Ventas,
      Descuento = Ventas_Nuevas * descuento_pct,
      Facturacion_Despues_Descuento = Ventas_Nuevas - Descuento,
      Costo_Ventas_Actual = Ventas_Actuales * costo_ventas_pct_actual,
      Costo_Ventas_Nuevo = Ventas_Nuevas * costo_ventas_pct_nuevo,
      Margen_Bruto_Actual = Ventas_Actuales - Costo_Ventas_Actual,
      Margen_Bruto_Nuevo = Facturacion_Despues_Descuento - Costo_Ventas_Nuevo,
      Diferencia_Margen_Bruto = Margen_Bruto_Nuevo - Margen_Bruto_Actual
    ) %>%
    summarise(
      Total_Margen_Bruto_Actual = sum(Margen_Bruto_Actual),
      Total_Margen_Bruto_Nuevo = sum(Margen_Bruto_Nuevo),
      Diferencia_Total_Margen_Bruto = sum(Diferencia_Margen_Bruto)
    )
}

# Crear un data frame con diferentes combinaciones
escenarios <- expand_grid(
  Incremento_Ventas_Pct = seq(0.10, 0.50, by = 0.05),
  Descuento_Pct = seq(0.01, 0.05, by = 0.01),
  Costo_Ventas_Pct_Nuevo = c(0.70, 0.68, 0.65)
)

# Añadir columnas para almacenar los resultados
resultados <- escenarios %>%
  rowwise() %>%
  mutate(
    Resultado = list(probar_escenario(Incremento_Ventas_Pct, Descuento_Pct, 0.70, Costo_Ventas_Pct_Nuevo))
  ) %>%
  unnest(Resultado, names_repair = "unique") # Evita duplicación de nombres

# Formatear las columnas numéricas con comas y evitar notación científica
resultados <- resultados %>%
  mutate(
    Total_Margen_Bruto_Actual = comma(Total_Margen_Bruto_Actual, big.mark = ","),
    Total_Margen_Bruto_Nuevo = comma(Total_Margen_Bruto_Nuevo, big.mark = ","),
    Diferencia_Total_Margen_Bruto = comma(Diferencia_Total_Margen_Bruto, big.mark = ",")
  )

# Convertir los porcentajes a formato de porcentaje
resultados <- resultados %>%
  mutate(
    Incremento_Ventas_Pct = percent(Incremento_Ventas_Pct),
  )
```

```

    Descuento_Pct = percent(Descuento_Pct),
    Costo_Ventas_Pct_Nuevo = percent(Costo_Ventas_Pct_Nuevo)
  )

# Seleccionar y reorganizar las columnas para mostrar los resultados
resultados_mostrados <- resultados %>%
  select(
    Incremento_Ventas_Pct,
    Descuento_Pct,
    Costo_Ventas_Pct_Nuevo,
    Total_Margen_Bruto_Actual,
    Total_Margen_Bruto_Nuevo,
    Diferencia_Total_Margen_Bruto
  )

# Mostrar los escenarios donde la diferencia en margen bruto es positiva y ordenados
resultados_mostrados %>%
  filter(as.numeric(gsub(",", "", Diferencia_Total_Margen_Bruto)) > 0) %>%
  arrange(desc(as.numeric(gsub(",", "", Diferencia_Total_Margen_Bruto)))) %>%
  print()

## # A tibble: 128 x 6
##   Incremento_Ventas_Pct Descuento_Pct Costo_Ventas_Pct_Nuevo
##   <chr>                <chr>          <chr>
## 1 50%                  1%            65%
## 2 50%                  2%            65%
## 3 45%                  1%            65%
## 4 50%                  3%            65%
## 5 45%                  2%            65%
## 6 40%                  1%            65%
## 7 50%                  1%            68%
## 8 50%                  4%            65%
## 9 45%                  3%            65%
## 10 40%                 2%            65%
## # i 118 more rows
## # i 3 more variables: Total_Margen_Bruto_Actual <chr>,
## #   Total_Margen_Bruto_Nuevo <chr>, Diferencia_Total_Margen_Bruto <chr>

```