MDPI

Article

ProyectoTD2025

1. Introducción

Los datos utilizados en este analisis, provienen de tikets de compra de Mercadona, dando una visión detallada de los habitos de consumo, tipos de productos variaciones de precios y patrones de compra a lo largo del tiempo. El objetivo principal de este trabajo es explorar y analizar las decisiones de compra registradas, con el fin de identificar tendencias y comportamientos que ayuden a comprender mejor las dinámicas de comsumo. A través de este análisis, se busca ofrecer una visión más detallada de los hábitos de consumo reflejados en los tickets, con posibles aplicaciones tanto en el estudio del comportamiento del consumidor como como para la toma de decisiones comerciales.

Cargamos las librerias necesarias:

Hemos cambiado los nombres a los archivos para que no den error con el siguiente codigo pero lo dejo como comentario para no sobreescribir los archivos al ejecutar mas de una vez:

Obtener lista de archivos PDF en la carpeta archivos <- list.files(path = "./data" , pattern = "\.pdf$", full.names = FALSE, ignore.case = TRUE)
Renombrar archivos secuencialmente for (i in seq_along(archivos)) { nombre_actual <- file.path(ruta_carpeta, archivos[i]) extension <- file_ext(archivos[i]) nombre_nuevo <- file.path(ruta_carpeta, paste0("M", i, ".", extension))
file.rename(from = nombre_actual, to = nombre_nuevo) }

2. Importación de los datos

cargamos los archivos de la carpeta

```
# Obtener lista de archivos PDF en la carpeta
archivos <- list.files(path =  "./data" , pattern = "\\.pdf$", full.names = TRUE,
#print(archivos)
```

Definimos vectores para almacenar los datos de los tickets

```
comercio <- c() #nombre del comercio
empresa <- c() #tipo y código de empresa
direccion <- c()
cp <- c() #código postal
telefono <- c() #Misma línea, tendremos que separar estos valores
fecha <- c()
hora <- c()
op <- c()
fs <- c()#factura
productos <- c() #lista con los productos comprados
```

Received:
Revised:
Accepted:
Published:

Citation: . ProyectoTD2025. *Journal Not Specified* **2023**, *1*, 0. https://doi.org/

```
total <- c() #total de la compra
forma_pago <- c()

#IVA
base_imp <- c() #Base imponible
cuota <- c() #Cuota
```

Extraemos la informacion de cada ticket y lo añadimos al vector correspondiente:

```
for (archivo in archivos) {

  pdf <- pdf_text(archivo) #leemos le archivo pdf
  ticket <- trimws(strsplit(pdf,split = "\n")[[1]]) #separamos por líneas
  ticket <- ticket[grep(".", ticket)] #quitamos las líneas vacías

  #procesamos los datos del ticket
  linea_comercio <- ticket[1]
  linea_direccion <- ticket[2]
  linea_cp <- ticket[3]
  linea_telefono <- ticket[4]
  linea_fecha_hora_op <- ticket[5]
  linea_fs <- ticket[6]
  p = 8
  linea_productos <- ticket[p]
    #unimos todos los productos en un solo caracter
  while (ticket[p+1] != ticket[grep("TOTAL", ticket)[1]]){
    p = p + 1
    linea_productos <- paste(linea_productos, ticket[p],sep = ";")
  }
  linea_total <- ticket[grep("TOTAL", ticket)[1]]
  linea_forma_pago <- ticket[p+2]
  linea_iva <- ticket[grep("TOTAL", ticket)[2]]

  #extaremos los datos
  com <- strsplit(linea_comercio,", ")[[1]]
  comercio <- c(comercio, com[1])
  empresa <- c(empresa, com[2])
  direccion <- c(direccion, trimws(linea_direccion))
  cp_info <- strsplit(trimws(linea_cp), " ")[[1]]
  cp <- c(cp, cp_info[1])
  telefono <- c(telefono, trimws(gsub("TELÉFONO:", "", linea_telefono)))
  fecha_hora_op <- strsplit(trimws(linea_fecha_hora_op), " ")[[1]]
  fecha_hora_op <- fecha_hora_op[grep(".", fecha_hora_op)]
  fecha <- c(fecha, fecha_hora_op[1])
  hora <- c(hora, fecha_hora_op[2])
  op <- c(op, gsub("OP:", "", fecha_hora_op[4]))
  fs <- c(fs, gsub("FACTURA SIMPLIFICADA:", "", linea_fs))
  productos <- c(productos, linea_productos)
  total <- c(total, trimws(gsub("TOTAL [(]€[)]", "", linea_total)))
  formapago <- strsplit(linea_forma_pago," ")[[1]]
  forma_pago <- c(forma_pago, paste0(formapago[1],formapago[2]))
```

```
  base_couta <- strsplit(trimws(linea_iva),split = " ")[[1]]
  base_couta <- base_couta[grep(".", base_couta)]
  base_imp <- c(base_imp,base_couta[2])
  cuota <- c(cuota, base_couta[3])

}
```

Creamos un data frame con los datos:

```
df <- data.frame(comercio, empresa, direccion, cp, telefono, fecha, hora,
                 op, fs, productos, total, forma_pago, base_imp, cuota,
                 stringsAsFactors = FALSE)

#Modificamos las clases de los datos
df$fecha <- as.Date(df$fecha,format = "%d/%m/%Y")
df$anio <- as.numeric(format(df$fecha, "%Y"))
df$mes <- as.numeric(format(df$fecha, "%m"))
df$dia <- as.numeric(format(df$fecha, "%d"))
# Elimina la columna fecha
df$fecha <- NULL

# Separar la columna fs en tres partes
fs_split <- strsplit(df$fs, "-")
# Crear las nuevas columnas a partir del resultado
df$num_tienda <- sapply(fs_split, function(x) x[1])
df$num_caja <- sapply(fs_split, function(x) x[2])
df$num_ticket <- sapply(fs_split, function(x) x[3])
# Eliminar la columna original fs
df$fs <- NULL
df$comercio <- NULL
df$empresa <- NULL


df$total <- as.numeric(gsub(pattern = ",",replacement = ".",df$total))
df$base_imp <- as.numeric(gsub(pattern = ",",replacement = ".",df$base_imp))
df$cuota <- as.numeric(gsub(pattern = ",",replacement = ".",df$cuota))
```

si hay aparcamiento o no

```
aparcamientos <- c()
for (archivo in archivos) {
  pdf <- pdf_text(archivo)
  ticket <- trimws(strsplit(pdf, split = "\n")[[1]])
  aparcamiento_lineas <- grep("Aparcamiento", ticket, value = TRUE)
  if (length(aparcamiento_lineas) > 0) {
    aparcamientos <- c(aparcamientos, paste(aparcamiento_lineas, collapse = " "))
  } else {
    aparcamientos <- c(aparcamientos, NA)
  }
}
```

```
df_aparcamiento <- data.frame(archivo = archivos, aparcamiento = aparcamientos, st
head(df_aparcamiento)
```

```
##           archivo aparcamiento
## 1   ./data/M1.pdf           NA
## 2  ./data/M10.pdf           NA
## 3 ./data/M100.pdf           NA
## 4 ./data/M101.pdf           NA
## 5 ./data/M102.pdf           NA
## 6 ./data/M103.pdf           NA
```

3. Analizamos los productos

En el df estan toda la informacion relativa a los productos en la columna producto la separamos en un nuevo dataframe con df_producto

```
df_producto <- df %>% select(c(num_ticket, productos)) %>%
  separate_rows(productos, sep = ";")
```

3.1. Procesamiento de pescado por kg

Primero nos encargamos de los productos de pesacdao que aparecen con este format: - Primera fila del pescado: Solo dice "PESCADO" - Segunda fila: Nombre del producto - Tercera fila: Detalles del precio

```
# Identificar filas con "PESCADO"
filas_pescado <- which(df_producto$productos == "PESCADO")

# Inicializar vectores para almacenar datos
num_ticket_vec <- character()
nombre_producto_vec <- character()
peso_kg_vec <- numeric()
precio_kg_vec <- numeric()
precio_total_vec <- numeric()

for (i in seq_along(filas_pescado)) {
  idx <- filas_pescado[i]

  # Extraer información básica
  num_ticket <- df_producto$num_ticket[idx]
  nombre <- df_producto$productos[idx + 1]  # Nombre en la siguiente fila

  # Procesar la fila de detalles
  detalles <- df_producto$productos[idx + 2]

  # Limpiar y dividir la cadena de detalles
  detalles_limpio <- gsub(",", ".", detalles)  # Reemplazar comas por puntos
  detalles_split <- strsplit(trimws(detalles_limpio), "\\s+")[[1]]

  # Extraer valores (asumiendo orden: peso, unidad, precio_kg, moneda, precio_tota
  if (length(detalles_split) >= 5) {
    peso_kg <- as.numeric(detalles_split[1])
```

```
    precio_kg <- as.numeric(detalles_split[3])
    importe <- as.numeric(detalles_split[5])

    # Almacenar en vectores
    num_ticket_vec <- c(num_ticket_vec, num_ticket)
    nombre_producto_vec <- c(nombre_producto_vec, nombre)
    peso_kg_vec <- c(peso_kg_vec, peso_kg)
    precio_kg_vec <- c(precio_kg_vec, precio_kg)
    precio_total_vec <- c(precio_total_vec, importe)
  }
}

df_pescado <- data.frame(
  num_ticket = num_ticket_vec,
  nombre = nombre_producto_vec,
  peso_kg = peso_kg_vec,
  precio_kg = precio_kg_vec,
  importe = precio_total_vec,
  stringsAsFactors = FALSE
)
```

3.2. Procesamiento fruta y la verdura

Borrar primero las filas de pescado para asegurarte de que los productos restantes vendidos por kg sean exclusivamente fruta y verdura.

```
#Identificar TODOS los bloques de pescado (3 filas cada uno)
bloques_pescado <- which(df_producto$productos == "PESCADO")

# Crear vector con TODAS las filas a eliminar (cada bloque son 3 filas)
filas_a_eliminar <- unlist(lapply(bloques_pescado, function(x) x:(x+2)))

# Eliminar todos los bloques
df_sin_pescado <- df_producto[-filas_a_eliminar, ]
```

Ahora hacemos el df_fruta_verdura

```
ind_detalles_kg <- grep("kg.*€/kg", df_sin_pescado$productos, value = FALSE)

df_fruta_verdura <- data.frame(
  num_ticket = df_sin_pescado$num_ticket[ind_detalles_kg],
  nombre = df_sin_pescado$productos[ind_detalles_kg - 1],
  detalles = df_sin_pescado$productos[ind_detalles_kg],
  stringsAsFactors = FALSE
) %>%
mutate(
  # Limpiar el nombre (eliminar números iniciales)
  nombre = gsub("^\\d+\\s*", "", nombre),

  # Extraer peso (kg) - primer número en la línea
  peso_kg = as.numeric(gsub(",", ".", str_extract(detalles, "^[0-9,]+"))),
```

```
  # Extraer precio por kg - método mejorado
  precio_kg = as.numeric(gsub(",", ".",
    str_extract(detalles, "[0-9,]+(?=\\s*€/kg)"))),

  # Extraer importe total - último número en la línea
  importe = as.numeric(gsub(",", ".",
    str_extract(detalles, "[0-9,]+$")))
) %>%
select(-detalles)
```

3.3. Procesamiento resto de productos sin kg

Extraemos cantidad descripcion y precio

Primero borramos los datos ya presentes en el df fruta y verdura

```
# Identificar las filas de detalles (kg y €/kg)
ind_detalles_kg <- grep("kg.*€/kg", df_sin_pescado$productos, value = FALSE)

# Las filas de nombres están justo antes de los detalles
ind_nombres_kg <- ind_detalles_kg - 1

# Combinar todos los índices a eliminar
filas_fruta_verdura <- sort(unique(c(ind_nombres_kg, ind_detalles_kg)))

# Eliminar filas ya procesadas
df_resto <- df_sin_pescado[-filas_fruta_verdura, ]
```

Creamos el dataframe de productos vendidios por unidades

```
df_productos_unidades <- df_resto %>%
  mutate(
    # 1. Extraer cantidad (siempre es el primer número)
    cantidad = as.numeric(str_extract(productos, "^\\d+")),

    # 2. Extraer posible precio en el nombre (para productos de 1 unidad)
    precio_en_nombre = ifelse(cantidad == 1,
                              as.numeric(gsub(",", ".", str_extract(productos, "\\d+
                              NA_real_),

    # 3. Procesamiento vectorizado de componentes
    componentes = strsplit(productos, "\\s+"),

    # 4. Extraer importe normal (para productos con múltiples unidades)
    importe_normal = sapply(componentes, function(x) {
      if(length(x) >= 3) as.numeric(gsub(",", ".", x[length(x)])) else NA_real_
    }),

    # 5. Determinar el importe final
    importe = ifelse(!is.na(precio_en_nombre), precio_en_nombre, importe_normal),

    # 6. Extraer descripción limpia (MODIFICACIÓN CLAVE)
```

```
    nombre = mapply(function(comp, prod, cant, precio_nombre) {
      # Primero eliminar la cantidad inicial (si existe)
      nombre_limpio <- gsub("^\\d+\\s*", "", prod)

      if(length(comp) <= 2) return(nombre_limpio)  # Caso simple

      if(!is.na(precio_nombre)) {
        # Para productos de 1 unidad: eliminar precio final
        gsub("\\s+\\d+,\\d+$", "", nombre_limpio)
      } else {
        # Para múltiples unidades: eliminar elementos numéricos finales
        paste(comp[2:(length(comp)-2)], collapse=" ")
      }
    }, componentes, productos, cantidad, precio_en_nombre, SIMPLIFY = TRUE) %>%
      str_trim()  # Eliminar espacios sobrantes
  ) %>%
  mutate(
    # 7. Calcular precio unitario
    precio_unitario = importe / cantidad
  ) %>%
  select(num_ticket, nombre, cantidad, precio_unitario, importe)
```

```
## Warning: There were 33 warnings in `mutate()`.
## The first warning was:
## i In argument: `importe_normal = sapply(...)`.
## Caused by warning in `FUN()`:
## ! NAs introducidos por coerción
## i Run `dplyr::last_dplyr_warnings()` to see the 32 remaining warnings.
```

3.4. dataframe final con todos los productos analizados:

```
# Añadir columna 'tipo' a cada dataframe
df_pescado <- df_pescado %>% mutate(tipo = "pescado")
df_fruta_verdura <- df_fruta_verdura %>% mutate(tipo = "fruta_verdura")
df_productos_unidades <- df_productos_unidades %>% mutate(tipo = "unidades")

# Unificar columnas para combinar
df_final <- bind_rows(
  df_pescado %>% select(num_ticket, nombre, cantidad = peso_kg, precio = precio_kg
  df_fruta_verdura %>% select(num_ticket, nombre, cantidad = peso_kg, precio = pre
  df_productos_unidades %>% select(num_ticket, nombre = nombre, cantidad, precio =
)

# Crear columna tiene_aparcamiento
if ("aparcamiento" %in% colnames(df_aparcamiento)) {
  df_aparcamiento <- df_aparcamiento %>%
    mutate(nombre_archivo = basename(archivo)) %>%
    mutate(num_ticket = str_extract(nombre_archivo, "\\d+")) %>%
    mutate(tiene_aparcamiento = !is.na(aparcamiento)) %>%
    select(num_ticket, tiene_aparcamiento)
```

```r
  # Convertir num_ticket a carácter en ambos dataframes
  df_final <- df_final %>% mutate(num_ticket = as.character(num_ticket))
  df_aparcamiento <- df_aparcamiento %>% mutate(num_ticket = as.character(num_tick

  # Combinar los datos
  df_final <- left_join(df_final, df_aparcamiento, by = "num_ticket")
} else {
  df_final$tiene_aparcamiento <- NA  # columna vacía si no existe 'aparcamiento'
}

# Resultado final ordenado
df_final <- df_final %>% arrange(num_ticket)

# Verificación final
cat("\nResumen del DataFrame final:\n")
```

```
##
## Resumen del DataFrame final:
```

```r
cat("- Pescado:", sum(df_final$tipo == "pescado"), "registros\n")
```

```
## - Pescado: 28 registros
```

```r
cat("- Fruta/Verdura:", sum(df_final$tipo == "fruta_verdura"), "registros\n")
```

```
## - Fruta/Verdura: 395 registros
```

```r
cat("- Unidades:", sum(df_final$tipo == "unidades"), "registros\n")
```

```
## - Unidades: 4702 registros
```

```r
cat("- Total:", nrow(df_final), "registros\n")
```

```
## - Total: 5125 registros
```

```r
print(head(df_final, 10))
```

```
##    num_ticket              nombre cantidad precio importe          tipo
## 1      007267    PERA CONFERENCIA    0.854   2.55    2.18 fruta_verdura
## 2      007267    MANZ. ROJA DULCE    1.474   2.90    4.27 fruta_verdura
## 3      007267      PIZZA BOLOÑESA    1.000   2.60    2.60      unidades
## 4      007267    MIX FRUTOS ROJOS    1.000   1.66    1.66      unidades
## 5      007267   PISTO DE VERDURAS    1.000   2.30    2.30      unidades
## 6      007267    FILETE DE TRUCHA    1.000   3.92    3.92      unidades
## 7      007267        PAN SEMILLAS    1.000   1.60    1.60      unidades
## 8      007267         SNACK PIPAS    1.000   1.40    1.40      unidades
## 9      007267 +PROT NATILLA VAINI    1.000   1.75    1.75      unidades
## 10     007267     SNACK CHOCOLATE    1.000   1.50    1.50      unidades
##    tiene_aparcamiento
## 1                  NA
```

```
## 2                NA
## 3                NA
## 4                NA
## 5                NA
## 6                NA
## 7                NA
## 8                NA
## 9                NA
## 10               NA
```

#Preguntas

¿Cuáles son los productos menos vendidos por unidades? ¿Y por kilos?

¿Qué productos han generado mayor ingreso total (precio × cantidad)?

¿Cuáles productos han aumentado o disminuido su venta a lo largo del tiempo?

¿Qué productos se compran habitualmente juntos?

¿Existen diferencias de precios para el mismo producto en diferentes tiendas o ubicaciones?

¿Qué días de la semana hay más ventas? ¿Y a qué horas?

¿En qué meses se venden más frutas/verduras, o pescados?

¿Desde qué ciudades se emiten más tickets?

¿Existen diferencias de consumo por ciudad?

¿Cuánto se gasta semanal o mensualmente en un supermercado?

¿Influye la disponibilidad de aparcamiento en el importe total de la compra?