Actividad 1: Mod. Estad. para la Toma Decisiones

Cristian Sarmiento

2024-08-07

1. Informe

1.1. Introducción

El mercado de viviendas urbanas es un sector complejo y dinámico que se encuentra en constante evolución. Para tener éxito en este mercado, las empresas inmobiliarias necesitan comprender en profundidad las tendencias del mercado, las necesidades de los clientes y la competencia.

En este informe, se presenta un análisis del mercado de viviendas urbanas en Cali. El análisis se basa en información detallada sobre diversas propiedades residenciales disponibles en el mercado de la ciudad de Cali.

El presente informe busca comprender los factores que determinan el precio de una vivienda con ciertas características definidas. De esta forma, la idea es adentrarse en el análisis del precio de viviendas estrato 4 con un área construida inferior o igual a 200 metros cuadrados, explorando la relación entre el precio y el espacio que ofrecen.

A través de un conjunto de datos denominado "vivienda", que alberga información detallada sobre diversas viviendas, este estudio busca identificar la relación entre el precio de la vivienda y su área construida mediante un modelo de regresión simple. Este análisis permitirá comprender cómo el tamaño de una vivienda impacta en su valor, brindando información valiosa para la toma de decisiones inmobiliarias.

2. Anexos

2.1 Preprocesamiento de información

Se realiza un primer análisis exploratorio de la base de datos. Este análisis se estructurará en la identificación de las columnas y dimensiones de la tabla, lo segundo será identificar los tipos de datos de las columnas y seleccionar cuáles serán relevantes para el análisis. Lo tercero será realizar la identificación de valores perdidos dentro de las columnas seleccionadas para el análisis y brindar posibles propuestas para poder solucionar los datos faltantes. Con la realización de estos pasos se puede determinar la relación de algunas variables con respecto a los objetivos del presente informe.

2.1 Cargue de base de datos

```
data(vivienda)
head(vivienda)
```

```
## # A tibble: 6 x 13
##      id zona    piso  estrato preciom areaconst parqueaderos banios habitaciones
##   <dbl> <chr>   <chr>   <dbl>   <dbl>     <dbl>        <dbl>  <dbl>        <dbl>
## 1  1147 Zona O~ <NA>        3     250        70            1      3            6
## 2  1169 Zona O~ <NA>        3     320       120            1      2            3
## 3  1350 Zona O~ <NA>        3     350       220            2      2            4
## 4  5992 Zona S~ 02          4     400       280            3      5            3
## 5  1212 Zona N~ 01          5     260        90            1      2            3
## 6  1724 Zona N~ 01          5     240        87            1      3            3
## # i 4 more variables: tipo <chr>, barrio <chr>, longitud <dbl>, latitud <dbl>
```

```
cat("Cantidad columnas: ", ncol(vivienda), "Cantidad filas: ", nrow(vivienda))
```

```
## Cantidad columnas:  13 Cantidad filas:  8322
```

2.2 Selección de datos relevantes

Según los objetivos del estudio, se identifican como variables relevantes zona, barrio, tipo como variables categóricas. Para análisis de variables numéricas se selecciona principalmente la variable preciom, como variables para análisis de caraterísticas se seleccionan piso, estrato, areaconst, banios, habitac. A continuación se realiza una verificación de datos categóricos relevantes:

```
unique_values <- unique(vivienda$zona)
print(paste0("Valores únicos para zona: ", paste0(unique_values, collapse = ", ")))
```

```
## [1] "Valores únicos para zona: Zona Oriente, Zona Sur, Zona Norte, Zona Oeste, Zona Centro, NA"
```

```
unique_values <- unique(vivienda$tipo)
print(paste0("Valores únicos para tipo: ", paste0(unique_values, collapse = ", ")))
```

```
## [1] "Valores únicos para tipo: Casa, Apartamento, NA"
```

```
# Generando vector para resumir las columnas seleccionadas
specific_columns <- c("preciom", "piso", "estrato", "areaconst", "banios", "habitac")
summary(vivienda)
```

```
##        id           zona               piso              estrato
##  Min.   :   1   Length:8322        Length:8322        Min.   :3.000
##  1st Qu.:2080   Class :character   Class :character   1st Qu.:4.000
##  Median :4160   Mode  :character   Mode  :character   Median :5.000
##  Mean   :4160                                         Mean   :4.634
##  3rd Qu.:6240                                         3rd Qu.:5.000
##  Max.   :8319                                         Max.   :6.000
##  NA's   :3                                            NA's   :3
##     preciom         areaconst       parqueaderos        banios
##  Min.   :  58.0   Min.   :  30.0   Min.   : 1.000   Min.   : 0.000
##  1st Qu.: 220.0   1st Qu.:  80.0   1st Qu.: 1.000   1st Qu.: 2.000
##  Median : 330.0   Median : 123.0   Median : 2.000   Median : 3.000
##  Mean   : 433.9   Mean   : 174.9   Mean   : 1.835   Mean   : 3.111
##  3rd Qu.: 540.0   3rd Qu.: 229.0   3rd Qu.: 2.000   3rd Qu.: 4.000
```

```
##  Max.   :1999.0   Max.   :1745.0   Max.   :10.000   Max.   :10.000
##  NA's   :2        NA's   :3        NA's   :1605     NA's   :3
##   habitaciones        tipo              barrio             longitud
##  Min.   : 0.000   Length:8322        Length:8322        Min.   :-76.59
##  1st Qu.: 3.000   Class :character   Class :character   1st Qu.:-76.54
##  Median : 3.000   Mode  :character   Mode  :character   Median :-76.53
##  Mean   : 3.605                                         Mean   :-76.53
##  3rd Qu.: 4.000                                         3rd Qu.:-76.52
##  Max.   :10.000                                         Max.   :-76.46
##  NA's   :3                                              NA's   :3
##     latitud
##  Min.   :3.333
##  1st Qu.:3.381
##  Median :3.416
##  Mean   :3.418
##  3rd Qu.:3.452
##  Max.   :3.498
##  NA's   :3
```

2.3 Limpieza de datos

Se evidencia que vienen valores de id con vacíos, para lo cual es necesario quitarlos de la base de datos:

```
faltantes_id = sum(is.na(vivienda$id))
cat("Cantidad de id faltantes:", faltantes_id)
```

```
## Cantidad de id faltantes: 3
```

```
## Se remueven los id vacios de la tabla:
vivienda = subset(vivienda, !is.na(id))
cat("Nueva cantidad filas: ", nrow(vivienda))
```

```
## Nueva cantidad filas:  8319
```

También se realiza la verificación de datos duplicados dentro de la tabla provista:

```
duplicates <- duplicated(vivienda)
n_duplicates <- sum(duplicates)
cat("Cantidad de filas duplicadas: ", n_duplicates)
```

```
## Cantidad de filas duplicadas:  0
```

```
## Se remueven los duplicados encontrados:
vivienda <- unique(vivienda)
cat("Nueva cantidad filas: ", nrow(vivienda))
```

```
## Nueva cantidad filas:  8319
```

Con lo identificado en la selección de variables se realiza la limpieza de las columnas de tipo, zona y barrio. Adicionalmente, al haber identificado un valor de “APTO” en la columna zona, esta se unifica con “APARTAMENTO” para normalizar los datos.

```r
vivienda$zona <- toupper(vivienda$zona)
vivienda$barrio <- toupper(vivienda$barrio)
vivienda$tipo <- toupper(vivienda$tipo)
vivienda$tipo <- ifelse(vivienda$tipo=='CASA','CASA','APARTAMENTO')
unique_values <- unique(vivienda$zona)
print(paste0("Valores únicos para zona: ", paste0(unique_values, collapse = ", ")))
```

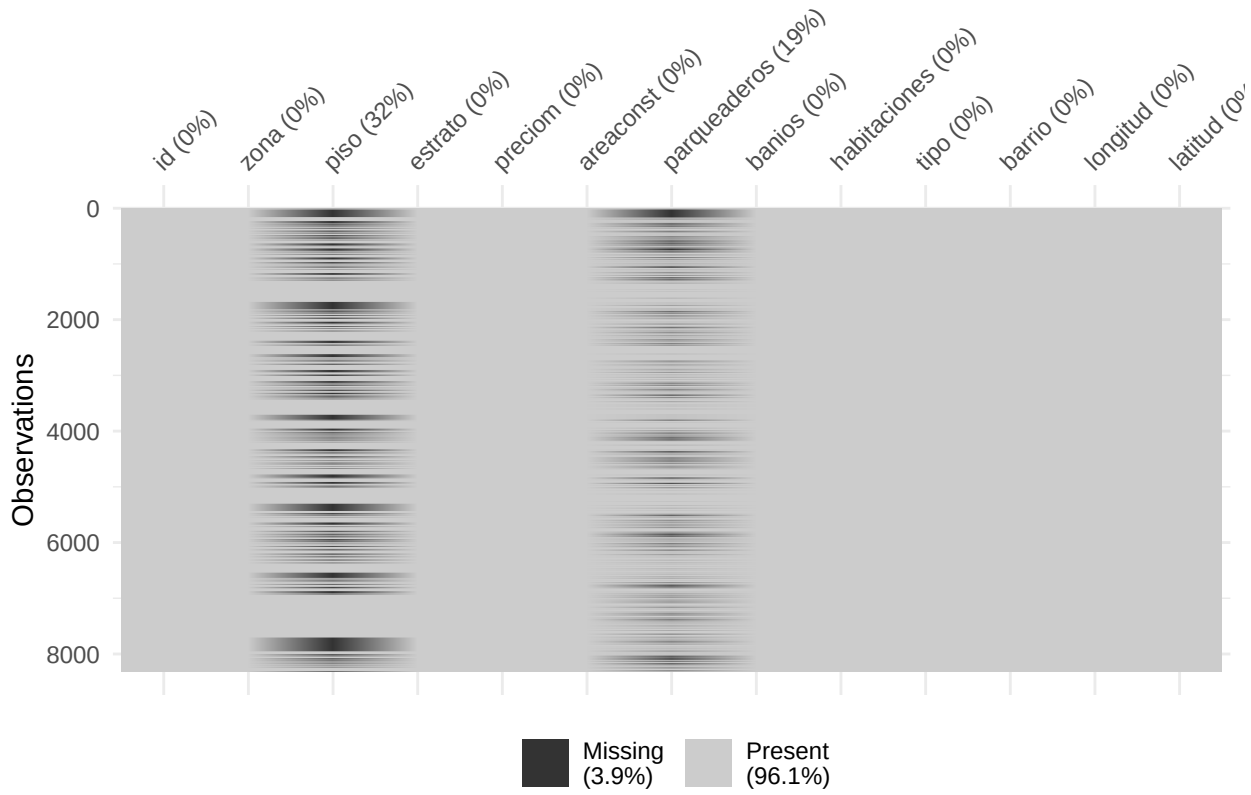
```
## [1] "Valores únicos para zona: ZONA ORIENTE, ZONA SUR, ZONA NORTE, ZONA OESTE, ZONA CENTRO"
```

```r
unique_values <- unique(vivienda$tipo)
print(paste0("Valores únicos para tipo: ", paste0(unique_values, collapse = ", ")))
```

```
## [1] "Valores únicos para tipo: CASA, APARTAMENTO"
```

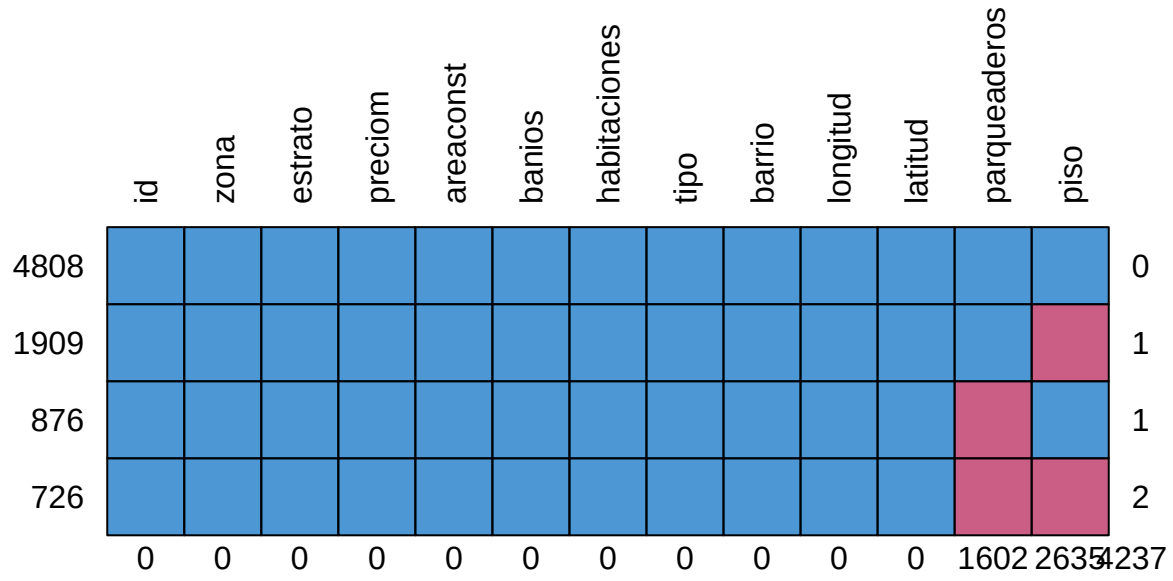
2.4 Identificación de faltantes

```r
## uses visdat
vis_miss(vivienda)
```



Haciendo uso de la librería `visdat` se logra identificar que dentro de la tabla dispuesta, después de la limpieza se tiene 2.9 % de datos faltantes. Todos distribuidos en las columnas `parqueaderos` y `piso`.

```
## uses visdat
md.pattern(vivienda, rotate.names = TRUE)
```




```
##      id zona estrato preciom areaconst banios habitaciones tipo barrio longitud
## 4808  1    1       1       1         1      1            1    1      1         1
## 1909  1    1       1       1         1      1            1    1      1         1
## 876   1    1       1       1         1      1            1    1      1         1
## 726   1    1       1       1         1      1            1    1      1         1
##       0    0       0       0         0      0            0    0      0         0
##      latitud parqueaderos piso
## 4808       1            1    1    0
## 1909       1            1    0    1
## 876        1            0    1    1
## 726        1            0    0    2
##            0         1602 2635 4237
```

Asi mismo, utilizando la librería `mice` se logra distinguir en qué categorías se encuentran mezclados los datos perdidos. Para el total de filas (8.319 filas) después de limpieza, se logra identificar que hay 1.602 (19,25 %) datos perdidos para la variable `parqueaderos`. Mientras que para la variable `piso` existen un total de 2.635 (31,67 %) filas sin datos.

En cuanto a la combinación de datos perdidos entre las dos variables identificadas, se resalta que hay 1.909 (22,94 %) filas con datos perdidos únicamente para la variable `piso`, hay 876 (10,53%) filas con datos perdidos únicamente para `parqueaderos` y 726 (8,72 %) filas que no tienen datos para ninguna de las dos columnas.

2.5 Posibles estrategías para mitigar datos faltantes

La primera posible solución para poder trabajar con los datos de la forma más completa posible puede ser precindir de usar las variables `parqueaderos` y `piso`. Esta estrategia tiene como ventaja que puede ser de las más rápidas de implementar, así como reducir la dimensionalidad de los datos, con lo que aplicar algunos algoritmos puede llegar a ser más eficiente. Así mismo, la reducción de información dentro de la tabla puede generar eliminación de información importante, lo que puede conllevar a presentar sesgo en el resultado.

```
vivienda <- vivienda[, -c("parqueaderos", "piso")]
```

La segunda solución puede ser implementar una imputación por alguna medida de tendencia central. Esta también es una medida simple de implementar y puede ayudar a conservar la dimensionalidad del conjunto de datos. Como contra puede que si los valores faltantes no son aleatorios o dependen de alguna categorización puede guiar a tener sesgo en los resultados.

`median_value <- median(vivienda$piso)` Se calcula la mediana y se imputa `vivienda$piso[is.na(vivienda$piso)] <- median_value`

También se puede intentar implementar una imputación por regresión. Esta técnica puede llegar a ser más sofisticada ya que tiene en cuenta la interacción entre variables. Sin embargo, este tipo de técnicas puede llegar a ser menos eficiente (tomar más tiempo de ejecución) que la imputación por medidas de tendencia central para conjuntos de datos grandes.