

TCVD-PRA2

Prueba

Francisco J. Morales & Antonio Martín

2022-05-27



# Contents

0.1	Descripción del dataset. . . . .	1
0.2	Integración y selección de los datos de interés a analizar. . . . .	1
0.3	Limpieza de los datos. . . . .	3
0.4	Análisis de los datos. . . . .	6
0.5	Representación de los resultados a partir de tablas y gráficas. . .	6
0.6	Resolución del problema. . . . .	6



## 0.1 Descripción del dataset.

Nos disponemos a estudiar un Dataset cedido por el MLIT (Ministry of Land, Infrastructure, Transport and Tourism) de Japón. Es un dataset que se publicó en Kaggle y contiene Un listado de transacciones de inmuebles desde 2005 a 2019 de las 7 prefecturas de Japón y puedes descargarse en la siguiente url:

<https://www.kaggle.com/datasets/nishiodens/japan-real-estate-transaction-prices>

Este dataset dispone no sólo del precio de venta del inmueble sino de variables cuantitativas como los metros cuadrados de algunas áreas a destacar, así como el total; como de variables cualitativas como el tipo del inmueble, la zona como su actividad (residencial o comercial), forma. Además podemos encontrar algunos flags de interés como si ha sido remodelado recientemente o si es excesivamente grande ( $>2000\text{m}^2$ ),

En este ejercicio, nos dispondremos a unificar todos los datos de las 47 prefecturas en el mismo dataset, hacer una limpieza de ellos y entrenar un modelo de regresión para valorar futuros inmuebles (AVM) que estén a la venta y compararlo con la misma oferta para tomar decisiones.

## 0.2 Integración y selección de los datos de interés a analizar.

Empezamos por cargar el primer fichero que encontramos. Observamos los diferentes tipos de activos que tenemos y descartamos para quedarnos únicamente con las viviendas. Para ello, nos basamos en los campos Use y Purpose. Nuestra filosofía que si alguno de los usos que se le da al inmueble es House, lo consideramos vivienda. Si el campo está vacío, tomamos el valor de Purpose.

```
df_japan <- read.csv("data/japan_housing_data/trade_prices/01.csv")
df_japan_0 <- read.csv("data/japan_housing_data/trade_prices/01.csv")
```

```
table(df_japan$Type)
```

```
##
##              Agricultural Land              Forest Land
##              23590              8682
##      Pre-owned Condominiums, etc. Residential Land(Land and Building)
##              20897              66809
##      Residential Land(Land Only)
##              66260
```

```
# df_japan <- df_japan[df_japan[, 'Type'] != 'Agricultural Land' & df_japan[, 'Type'] != 'Forest
```

```
head(table(df_japan$Use))
```

```
##
##              Factory
##              104989              375
##      Factory, Office      Factory, Office, Other
```

```
##                                151                                3
## Factory, Office, Parking Lot      Factory, Office, Shop
##                                2                                8

head(table(df_japan$Purpose))

##
##      Factory   House   Office   Other   Shop
## 129356     333   42410    1272   10539   1481

df_japan[df_japan$Use=='','Use'] <- df_japan[df_japan$Use=='','Purpose']
df_japan <- subset(df_japan, grepl("House", df_japan$Use))
```

Como desconocemos si cada vivienda tiene un identificador como la referencia catastral aquí en España, o algún indicador en el que se vea si se ha hecho una división horizontal. Es más, como ni siquiera disponemos de la dirección, no es imposible determinar al 100% si dos ventas se refieren al mismo inmueble. El hecho aquí, es que hablamos de ventas y suponemos que una venta está duplicada si todos los campos son iguales. Es decir, si se vende el mismo inmueble, el mismo año, en el mismo quarter, pero con diferente precio, lo vamos a considerar como una venta diferente. Si mas adelante vemos que esto empeora el modelo, rectificaremos.

```
duplicados <- nrow(df_japan[duplicated(df_japan), ])
df_japan <- df_japan[!duplicated(df_japan), ]
```

Una vez hemos cargado el primer fichero, hacemos un bucle para cargar el resto. Para optimizar recursos, iremos filtrando los tipos y revisando los duplicados en cada fichero.

```
resources_root <- "data/japan_housing_data/trade_prices/"

for(i in seq(from=2, to=47)){
  index_file <- paste('0', toString(i), sep = "", collapse = NULL)
  file <- paste(resources_root, substr(index_file, nchar(index_file)-1, nchar(index_file)),
  df <- read.csv(file)

  df[df_japan$Use=='','Use'] <- df[df$Use=='','Purpose']
  df <- subset(df, grepl("House", df$Use))

  duplicados <- nrow(df[duplicated(df), ]) + duplicados
  df <- df[!duplicated(df), ]

  df_japan <- union(df_japan, df)
}
```

Ahora, seleccionaremos los campos que a priori creemos que nos servirán para el modelo y excludiremos las redundantes.

```
columns <- c("No", "Type", "Region", "MunicipalityCode", "Prefecture", "Municipality", "DistanceToNearestStation", "TimeToNearestStation", "MaxTimeToNearestStation", "TradePrice", "FloorPlanArea", "LandShape", "Frontage", "BuildingYear", "Structure", "CityPlanning", "Year", "YearBuilt")
```

```
df_japan <- df_japan[,columns]
```

### 0.3 Limpieza de los datos.

Para empezar le daremos una vista general a set de datos.

```
str(df_japan)
```

```
## 'data.frame':    1695459 obs. of  22 variables:
## $ No                : int  2 3 5 9 10 11 12 13 14 19 ...
## $ Type              : chr  "Residential Land(Land Only)" "Pre-owned Condominiums, etc."
## $ Region            : chr  "Residential Area" "" "Residential Area" "Residential Area"
## $ MunicipalityCode  : int  1101 1101 1101 1101 1101 1101 1101 1101 1101 1101 ...
## $ Prefecture        : chr  "Hokkaido" "Hokkaido" "Hokkaido" "Hokkaido" ...
## $ Municipality      : chr  "Chuo Ward,Sapporo City" "Chuo Ward,Sapporo City" "Chuo Ward,Sapporo City"
## $ DistrictName      : chr  "Asahigaoka" "Asahigaoka" "Asahigaoka" "Asahigaoka" ...
## $ NearestStation    : chr  "Maruyamakoen" "Maruyamakoen" "Maruyamakoen" "Maruyamakoen"
## $ TimeToNearestStation : chr  "27" "20" "23" "29" ...
## $ MaxTimeToNearestStation: int  27 20 23 29 23 21 28 28 25 60 ...
## $ TradePrice        : num  3.8e+07 1.9e+07 2.5e+07 2.0e+07 1.4e+07 3.8e+06 2.5e+07 4.0e+07
## $ FloorPlan         : chr  "" "4LDK" "" "" ...
## $ Area              : int  310 95 430 165 90 60 580 400 80 185 ...
## $ UnitPrice         : num  120000 NA 58000 120000 NA NA NA NA NA NA ...
## $ LandShape         : chr  "Rectangular Shaped" "" "Rectangular Shaped" "Semi-square Shaped"
## $ Frontage          : num  21.5 NA 16 12.6 NA 5.1 50 14 NA 15 ...
## $ BuildingYear      : int  NA 1997 NA NA 1989 1954 1978 1989 2002 1997 ...
## $ Structure         : chr  "" "RC" "" "" ...
## $ CityPlanning      : chr  "Category I Exclusively Low-story Residential Zone" "Neighborhood Planning Zone"
## $ Year              : int  2018 2018 2018 2017 2017 2010 2008 2017 2010 2006 ...
## $ Quarter           : int  4 4 2 4 4 3 2 2 2 4 ...
## $ Renovation        : chr  "" "Not yet" "" "" ...
```

```
summary(df_japan)
```

```
##           No                Type                Region                MunicipalityCode
## Min.      :      1  Length:1695459  Length:1695459  Min.      : 1101
## 1st Qu.: 25886  Class :character  Class :character  1st Qu.:12228
## Median : 66074  Mode  :character  Mode  :character  Median :14153
## Mean   : 95151
## 3rd Qu.:142559
## Max.   :406575
##
## Prefecture      Municipality      DistrictName      NearestStation
## Length:1695459  Length:1695459  Length:1695459  Length:1695459
## Class :character  Class :character  Class :character  Class :character
## Mode  :character  Mode  :character  Mode  :character  Mode  :character
##
##
##
```

```
##
## TimeToNearestStation MaxTimeToNearestStation TradePrice
## Length:1695459 Min. : 0.00 Min. :1.000e+02
## Class :character 1st Qu.: 8.00 1st Qu.:1.100e+07
## Mode :character Median : 14.00 Median :2.100e+07
## Mean : 23.63 Mean :2.466e+07
## 3rd Qu.: 25.00 3rd Qu.:3.300e+07
## Max. :120.00 Max. :2.200e+10
## NA's :66944
## FloorPlan Area UnitPrice LandShape
## Length:1695459 Min. : 10.0 Min. : 1 Length:1695459
## Class :character 1st Qu.: 65.0 1st Qu.: 16000 Class :character
## Mode :character Median : 110.0 Median : 26000 Mode :character
## Mean : 155.7 Mean : 40192
## 3rd Qu.: 185.0 3rd Qu.: 50000
## Max. :5000.0 Max. :1100000
## NA's :1682516
## Frontage BuildingYear Structure CityPlanning
## Min. : 0.4 Min. :1945 Length:1695459 Length:1695459
## 1st Qu.: 7.4 1st Qu.:1985 Class :character Class :character
## Median :10.5 Median :1998 Mode :character Mode :character
## Mean :11.5 Mean :1996
## 3rd Qu.:14.0 3rd Qu.:2009
## Max. :50.0 Max. :2020
## NA's :655028 NA's :83422
## Year Quarter Renovation
## Min. :2005 Min. :1.000 Length:1695459
## 1st Qu.:2010 1st Qu.:2.000 Class :character
## Median :2013 Median :3.000 Mode :character
## Mean :2013 Mean :2.508
## 3rd Qu.:2016 3rd Qu.:3.000
## Max. :2019 Max. :4.000
##
```

```
duplicados
```

```
## [1] 0
```

Podemos ver que, según nuestra definición, no tenemos ventas duplicadas.

Tomamos las siguientes decisiones.

- Eliminamos el ID que realmente no nos dice nada.
- Nos quedamos con el máximo tiempo hasta la estación más próxima. Somos pesimistas en este aspecto.
- Eliminamos el precio en moneda extranjera.
- Calculamos nosotros el precio por metro cuadrado dividiendo el precio de la venta entre el área. Asumimos que el campo AREA, incluye el resto de campos referidos a superficies.

```
df_japan$Region[df_japan$Region == ''] <- "Other"
```



```
table(df_japan$Type,df_japan$Region)
```

```
##
##
##      Commercial Area Industrial Area   Other
## Agricultural Land           0           0    63
## Forest Land                 0           0     5
## Pre-owned Condominiums, etc. 0           0 572100
## Residential Land(Land and Building) 40158       2280     0
## Residential Land(Land Only)       438          13     0
```

```
##
##      Potential Residential Area
## Agricultural Land           0
## Forest Land                 0
## Pre-owned Condominiums, etc. 0
## Residential Land(Land and Building) 116
## Residential Land(Land Only)       12
```

```
##
##      Residential Area
## Agricultural Land           0
## Forest Land                 0
## Pre-owned Condominiums, etc. 0
## Residential Land(Land and Building) 1067794
## Residential Land(Land Only)       12480
```

```
table(df_japan$Year)
```

```
##
## 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015
## 14265 42639 94739 115128 124339 136435 125861 133040 144175 140217 147468
## 2016 2017 2018 2019
## 144923 134734 122305 75191
```

```
df_japan$FloorPlan[df_japan$FloorPlan == ''] <- "-"
```

```
df_japan$DistrictName[df_japan$DistrictName == '(No Address)'] <- "-"
```

```
df_japan$DistrictName[df_japan$DistrictName == ''] <- "-"
```

```
df_japan$NearestStation[df_japan$NearestStation == ''] <- "-"
```

```
df_japan$LandShape[df_japan$LandShape == ''] <- "-"
```

```
df_japan$Renovation[df_japan$Renovation == ''] <- "-"
```

```
df_japan$Structure[df_japan$Structure == ''] <- "-"
```

```
df_japan[is.na(df_japan$Frontage), 'Frontage'] <- 0
```

```
df_japan <- df_japan[!is.na(df_japan$TradePrice),]
```

```
df_japan <- df_japan[!is.na(df_japan$Area),]
```

```
df_japan$UnitPrice<- df_japan$TradePrice/df_japan$Area
```

```
#Outliers
```

Quitamos Outliers para limpiar los datos.

```
rating_plot <- ggplot(df_japan, aes(y=TradePrice)) + geom_boxplot()  
ggplotly(rating_plot)
```

```
## PhantomJS not found. You can install it with webshot::install_phantomjs(). If it i
```

```
out_ <- boxplot.stats(df_japan$TradePrice)$out  
idx_out_ <- which(df_japan$TradePrice %in% out_)  
df_japan <- df_japan[-idx_out_,]
```

## 0.4 Análisis de los datos.

0.4.1 Selección de los grupos de datos que se quieren analizar/comparar.

0.4.2 Comprobación de la normalidad y homogeneidad de la varianza.

0.4.3 Aplicación de pruebas estadísticas para comparar los grupos de datos.

0.5 Representación de los resultados a partir de tablas y gráficas.

0.6 Resolución del problema.