Japan Automated Validation Model

Tipología y ciclo de vida de los datos - Práctica 2

Francisco J. Morales & Antonio Martín

2022-05-30

Contents

Chapter 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 47 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 inmuble sino de variables cuantitativas como los metros cuadrados de algunas áreas a destacar, así como el total; como de varialbes 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 (>2000m2),

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.

Chapter 2

Integración y selección de datos

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
```

```
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)),'.csv',sep = "",collapse = NULL)
  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 excluiremos las redundantes.

```
columns <- c("No","Type","Region","MunicipalityCode","Prefecture","Municipality",
             "DistrictName","NearestStation","TimeToNearestStation",
             "MaxTimeToNearestStation","TradePrice","FloorPlan","Area","UnitPrice",
             "LandShape","Frontage","BuildingYear","Structure","CityPlanning",
             "Year","Quarter","Renovation")

df_japan <- df_japan[,columns]
```

Chapter 3

Limpieza de los datos

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

```
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                                        Mean   :19358
##  3rd Qu.:142559                                        3rd Qu.:27203
##  Max.   :406575                                        Max.   :47382
##
##   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.
- 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.
- Eliminamos los terrenos no edificados.
- Limpiamos datos nulos.

```
df_japan$Region[df_japan$Region == ''] <- "Other"
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$Area
df_japan <- df_japan[!is.na(df_japan$BuildingYear),]

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

```
##
##                                     Commercial Area Industrial Area   Other
##   Pre-owned Condominiums, etc.                    0               0  557357
##   Residential Land(Land and Building)         34429            2156       0
```

```
##
##                                     Potential Residential Area
##   Pre-owned Condominiums, etc.                                0
##   Residential Land(Land and Building)                       102
##
##                                     Residential Area
##   Pre-owned Condominiums, etc.                     0
##   Residential Land(Land and Building)        1017993
```

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

```
##
##   2005   2006   2007   2008   2009   2010   2011   2012   2013   2014   2015
##  13788  41545  92407 112551 118651 130856 119981 126467 136024 131875 138853
##   2016   2017   2018   2019
## 136477 126649 115185  70728
```

3.1 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 is installed, please
```

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

Chapter 4

Análisis de los datos

4.1 Selección de grupos

A priori, nos vamos a centrar en estimar el precio de la vivienda a partir del propio tipo. Vamos a suponer que el precio cuadrado de los pisos vendran dados por una distribución aleatoria con su media y varianza y estas serán diferentes de la distribución formada por los comercios, o chalets, etc.

4.2 Comprobación de normalidad y homocedasticidad

Al tener un volumen elevado de registros podemos aplicar el teorema central del límite por el que suponemos que la media muestral seguirá una distribución normal.

En cuanto a la homocedasticidad, podemos comprobar si la varianza se mantiene entre inmuebles de distinto tipo:

```
leveneTest(y = df_japan$TradePrice, group = df_japan$Type, center = "median")
```

```
## Levene's Test for Homogeneity of Variance (center = "median")
##            Df F value    Pr(>F)
## group       1   23554 < 2.2e-16 ***
##       1569712
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Así a partir del resultado del test (p-value $< 0{,}05$) podemos concluir que no se da una homogeneidad en la varianza entre los distintos tipos de inmuebles.

4.3 Aplicación de pruebas estadísticas

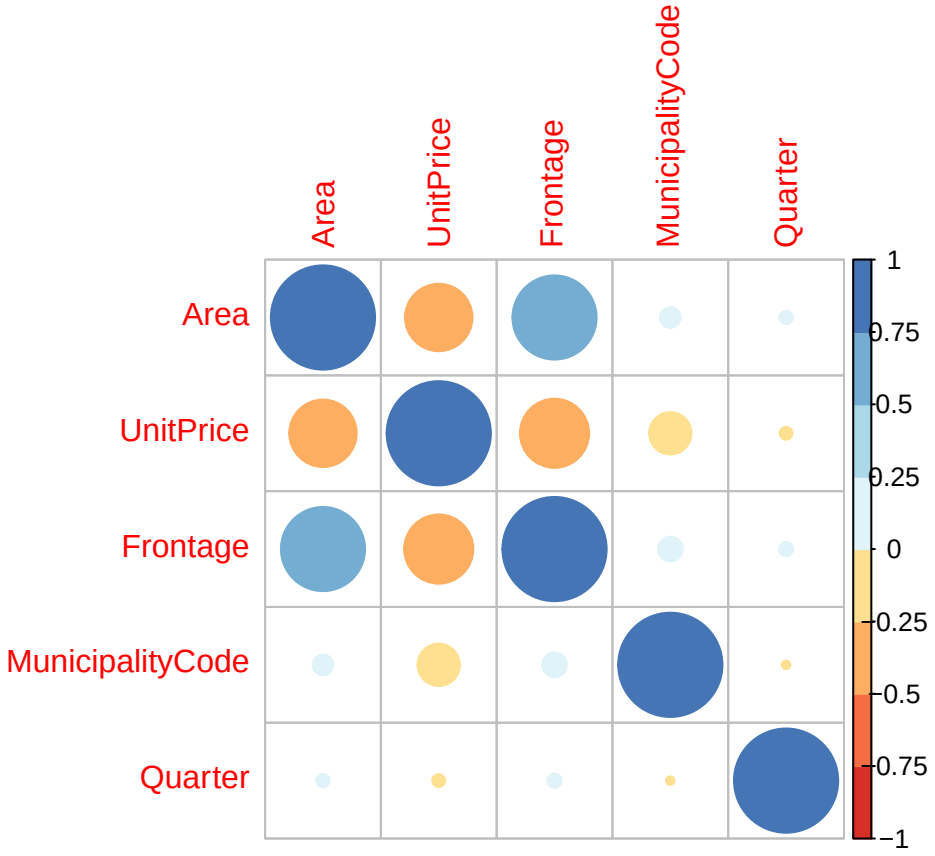
4.3.1 Correlaciones

Empezaremos primero analizando las correlaciones entre las variables numéricas.

```
cuantitativas <- df_japan %>% select (c(Area, UnitPrice, Frontage, MunicipalityCode, Quarter))

M <-cor(cuantitativas)

corrplot(M,
         col=brewer.pal(n=8, name="RdYlBu"))
```



Vemos que las variables Area y Frontage están correlacionadas entre sí y de manera más clara con UnitPrice que MunicipalityCode y Quarter, pero por sí solas no explican el precio por metro cuadrado. Tendremos que ver la influencia de las otras variables a lo largo del tiempo.

4.3.1.1 Year

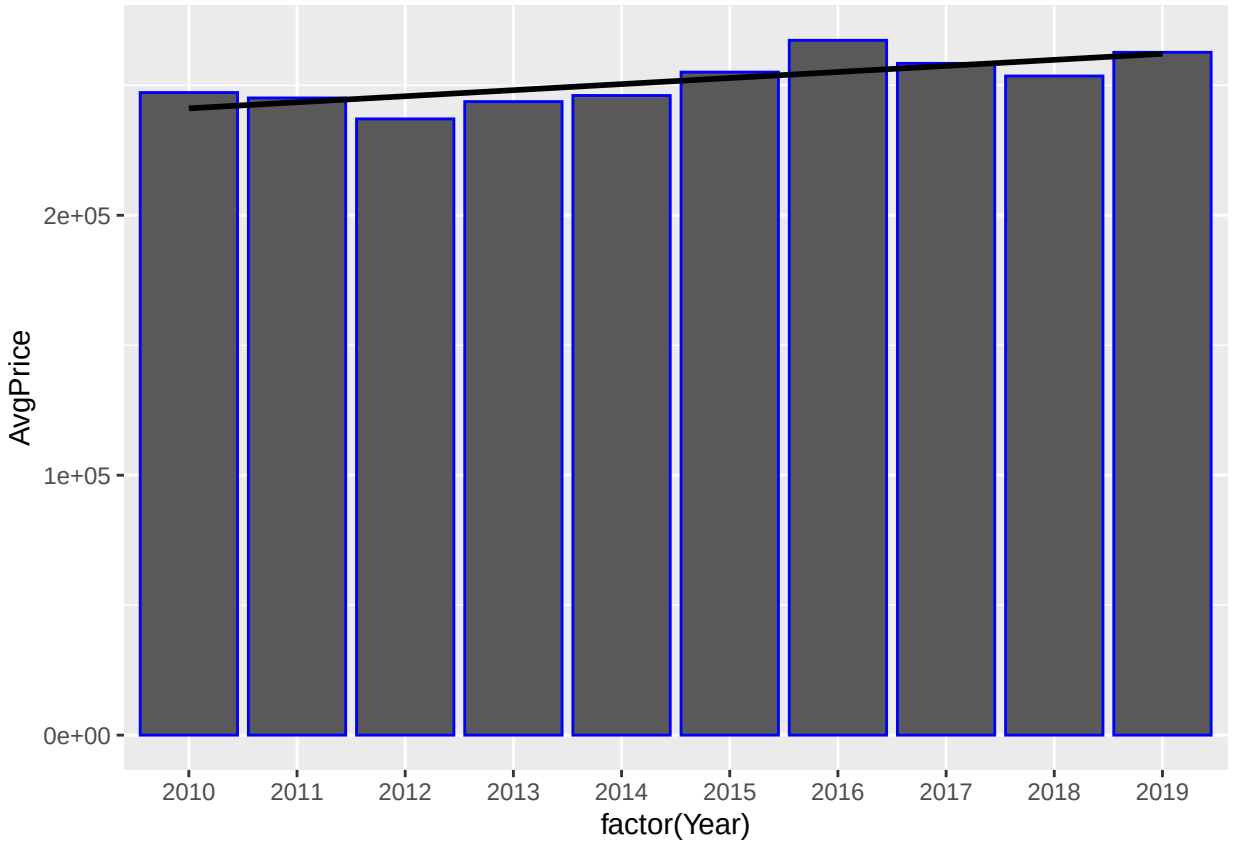
Veamos como ha evolucionado el metro cuadrado a lo largo del tiempo.

```
Unit.Price.Year <- df_japan %>% group_by(Year) %>% summarise(mean(UnitPrice))

colnames(Unit.Price.Year) <- c("Year","AvgPrice")

tb0 <- ggplot(Unit.Price.Year[Unit.Price.Year$Year>=2010,], aes(x=factor(Year), y=AvgPrice)) +
       geom_bar(col='blue',stat='identity') + geom_smooth(method = "lm", se=FALSE, color="black", aes(g
tb0
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Vemos que en 2012 hubo una bajada general, pero desde ese momento el valor del metro cuadrado ha ido creciendo. El punto aquí es que seguramente un el precio del metro de un piso no haya evolucionado igual que el de una casa en mitad del bosque. Veamos como a los largo de los años se comporta el precio del metro cuadrado en función de diferentes variables.

4.3.1.2 Type

Veamos respecto al Tipo.

```
Unit.Price.Year.Type <- df_japan %>% group_by(Year,Type) %>% summarise(mean(UnitPrice), .groups = 'drop'

colnames(Unit.Price.Year.Type) <- c("Year","Type","AvgPrice")

tb1 <- ggplot(Unit.Price.Year.Type[Unit.Price.Year.Type$Year>=2014,], aes(x=factor(Year), y=AvgPrice))+
        geom_bar(col='blue',stat='identity') + geom_smooth(method = "lm", se=FALSE, color="black", aes(g
tb1
```

```
## `geom_smooth()` using formula 'y ~ x'
```

Vemos aquí que no solo el precio es significativamente diferente, sino que crecen a diferentes velocidades.

4.3.1.3 CityPlanning

Veamos los datos respecto a las zonas.

```
Unit.Price.Year.Planning <- df_japan %>% group_by(Year,CityPlanning) %>% summarise(mean(UnitPrice), .gro

colnames(Unit.Price.Year.Planning) <- c("Year","CityPlanning","AvgPrice")

tb2 <- ggplot(Unit.Price.Year.Planning[Unit.Price.Year.Planning$Year>=2016,], aes(x=factor(Year), y=AvgP
        geom_bar(col='blue',stat='identity') + geom_smooth(method = "lm", se=FALSE, color="black", aes(g

tb2
```

```
## `geom_smooth()` using formula 'y ~ x'
```

Aquí vemos la misma situación que antes. No es lo mismo una vivienda en una zona comercial donde el precio es alto y se mantiene que en una zona industrial que cada vez se va devaluando que en una "Quasi-residential Zone" donde en estos últimos años empieza a valer bastante más el metro cuadrado.

4.3.1.4 Prefecture

Veamos respecto a la Prefectura.

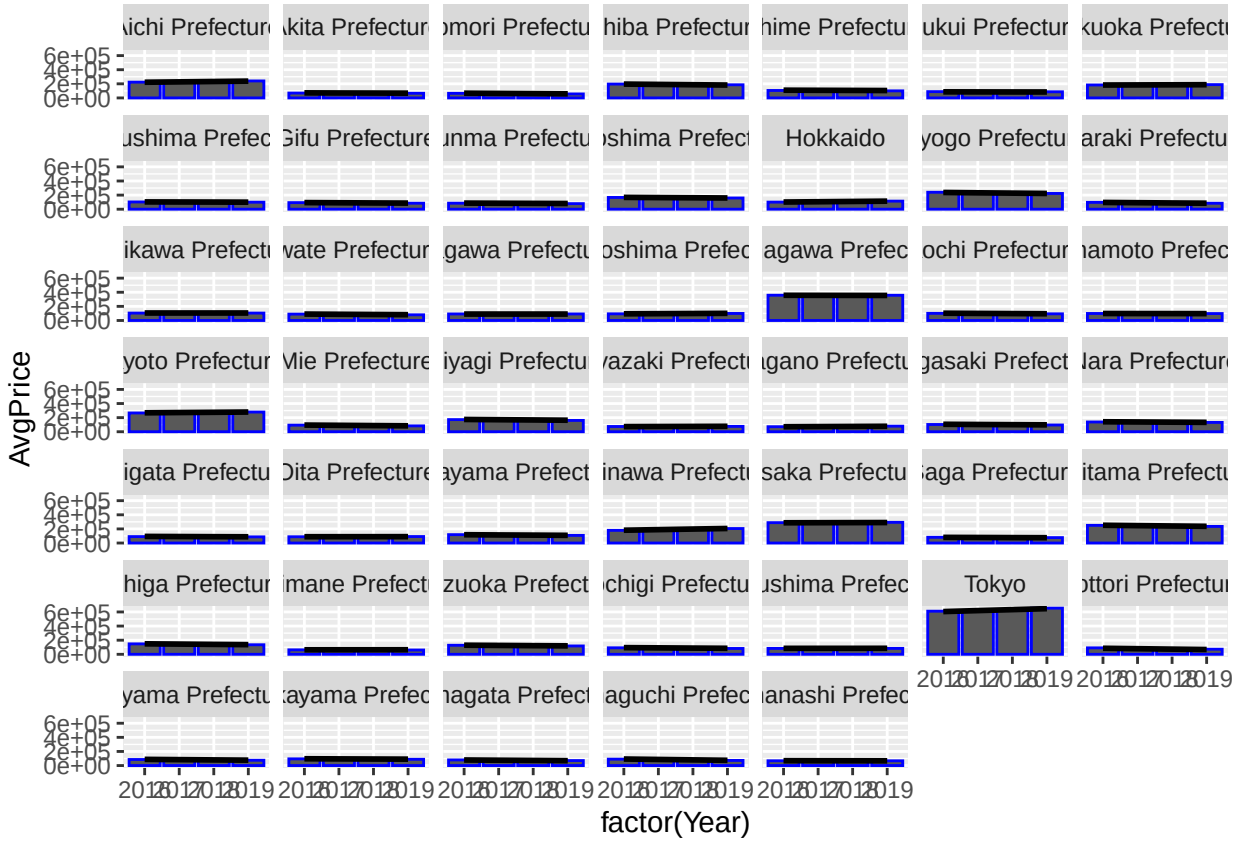
```
Unit.Price.Year.Prefecture <- df_japan %>% group_by(Year,Prefecture) %>% summarise(mean(UnitPrice), .gro

colnames(Unit.Price.Year.Prefecture) <- c("Year","Prefecture","AvgPrice")

tb3 <- ggplot(Unit.Price.Year.Prefecture[Unit.Price.Year.Prefecture$Year>=2016,], aes(x=factor(Year), y=
        geom_bar(col='blue',stat='identity') + geom_smooth(method = "lm", se=FALSE, color="black", aes(g

tb3
```

```
## `geom_smooth()` using formula 'y ~ x'
```



La misma situación es un factor a tener en cuenta también que no es lo mismo una vivienda en Tokio que en Hokkaido. Incluso, posiblemente dentro de la misma prefectura, la evolución del metro cuadrado sea diferente.

4.3.2 Comparación de medias

Pasemos a estudiar si, por lo general, el precio de un inmueble es diferente dependiendo del tipo. En este caso nos hemos centrado en los tipos más comunes que son: Pisos (Pre-owned Condominiums, etc.) y casas (Residential Land(Land and Building). Como hemos asumido normalidad y hemos comprobado que no tienen la misma varianza, debemos llevar a cabo la prueba t-Student para muestras independeientes con varianzas diferentes.

```
t.test(df_japan$TradePrice[which(df_japan$Type == "Pre-owned Condominiums, etc.")],
       df_japan$TradePrice[which(df_japan$Type == "Residential Land(Land and Building)")],
       var.equal=FALSE)
```

```
##
##  Welch Two Sample t-test
##
## data:  df_japan$TradePrice[which(df_japan$Type == "Pre-owned Condominiums, etc.")] and df_japan$Trade
## t = -153.49, df = 1244467, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -3513749 -3425144
## sample estimates:
```

```
## mean of x mean of y
##  20149066  23618513
```

Como el p-value es inferior a 0.05, descartamos la hipótesis nula de que las medias poblacionales de los diferentes tipos son iguales.

4.3.3 Modelo de regresión

En un principio se pensaba hacer un modelo para estimar el valor del metro cuadrado a traves de **comparables**. El hecho de cómo ha evolucionado el precio del metro cuadrado en los últimos años y en la hetereogeneidad de este en diferentes prefectura y dentro de estas, en los diferentes municipios y distritos nos impedía encontrar un inmueble similar a otro. Una opción con más tiempo sería aplicar un algoritmo de Clustering o un Árbol de decisión para esto. En este caso hemos optado en seleccionar las variables que hemos visto que influyen en el precio del metro cuadrado y crear un modelo de regresión lineal.

Como hemos observado que la situación ha cambiado con los años, siendo muy diferente de la actual, únicamente nos quedaremos con los datos de los 2 últimos años registrados en el dataset.

```
df_japan <- df_japan[df$Year>=2018,]
```

A continuación creamos la muestra aleatoria. Añadimos las variables que hemos estudiado y algunas que pueden tener relevancia para el diseño del modelo.

```
set.seed(20)
train_index <- sample(1:nrow(df_japan), 0.9 * nrow(df_japan))
test_index <- setdiff(1:nrow(df_japan), train_index)
df_japan_test <- df_japan[test_index,]

columns_model <- c("Type","Region",'MunicipalityCode',"Prefecture","MaxTimeToNearestStation",
                   "FloorPlan","Area","UnitPrice",
              "LandShape","Frontage","CityPlanning","Year","Quarter","Renovation", "Structure","BuildingY

df_japan_train <- df_japan[train_index,columns_model]
```

Creamos nuestro primer modelo, empleando únicamente las variables Area y Frontage que habíamos visto que presentan una fuerte correlación con el precio de las viviendas.

```
model <- lm(UnitPrice~Area+ Frontage, data=df_japan_train)
summary(model)$r.squared
```

```
## [1] 0.2217716
```

Como se puede ver, el valor de R^2 es bastante bajo por lo que planteamos añadir algunas de las variables categóricas del dataset (Type, Region, Prefecture, CityPlanning y Renovation) para observar si mejoran el modelo a crear:

```
model2 <- lm(UnitPrice~factor(Type)+Area + Frontage+factor(Region) + factor(Prefecture)+
factor(CityPlanning)+factor(Renovation), data=df_japan_train)
summary(model2)$r.squared
```

```
## [1] 0.5558014
```

Como podemos ver, tras añadir estas variables el modelo mejora bastante.

Añadimos también Structure y BuildingYear a la regresión con el siguiente resultado:

```
model3 <- lm(UnitPrice~factor(Type)+Area + Frontage+factor(Region)+ factor(Prefecture)+
factor(CityPlanning)+factor(Renovation)+ factor(Structure) + BuildingYear, data=df_japan_train, na.actio
summary(model3)$r.squared
```

```
## [1] 0.6547424
```

Esto mejora en gran medida la regresión, por lo que mantenemos este como modelo final.

Por último llevamos a cabo la predicción de los valores de TradePrice para los datos de test y mostramos también las diferencias entre los valores reales y predichos.

```
df_japan_test$NewValue <- predict(model3, df_japan_test, interval = c("confidence"), se.fit=FALSE)


df_japan_result <- df_japan_test[,c('No','Prefecture','TradePrice')]
df_japan_result$EstimateTradePrice <-df_japan_test$NewValue *df_japan_test$Area

summary(df_japan_result$TradePrice - df_japan_result$EstimateTradePrice)
```

```
##       fit                 lwr                 upr
##  Min.   :-414166088   Min.   :-400658502   Min.   :-427673673
##  1st Qu.:  -7669994   1st Qu.:  -7011753   1st Qu.:  -8358064
##  Median :   -953177   Median :   -403794   Median :  -1499139
##  Mean   :     91561   Mean   :    953364   Mean   :   -770242
##  3rd Qu.:   6185011   3rd Qu.:   6777465   3rd Qu.:   5589856
##  Max.   : 652922269   Max.   : 673225242   Max.   : 632619295
```

Chapter 5

Resolución del problema

Como conclusión tras el estudio del dataset podemos concluir que la estimación del valor exacto de un inmueble atiende a un gran número de variables, complicando en gran medida la generación de modelos predictivos. Aun así, el modelo generado es capaz de reducir el error cometido por la tasación de inmuebles únicamente por su tamaño (primer modelo) y pone de manifiesto la importancia de la ubicación, tipo y momento de construcción de las viviendas a estudiar.

Chapter 6

Generación del Output

```
write.csv(x = df_japan_result, file = "df_japan_result.csv", row.names = TRUE)
```

Chapter 7

Tabla de contribuciones

Contribuciones	Firma
Investigación previa	FJMH - AME
Redacción de las respuestas	FJMH - AME
Desarrollo código	FJMH - AME