Preprocesamiento y análisis de datos

Pablo Maroto López

12-2020

Índice

Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende responder?

El dataset *Mobile Price Classification* contiene los datos de venta de distintos móviles. Los datos consisten en las características de móbiles y el rango de precio de su valor de venta.

Con este conjunto de datos se puede determinar qué variables infuyen más en el precio final del móvil, si existe algún tipo de relación entre variables o establecer un modelo para predecir el precio de un móvil a partir de sus características.

Integración y selección de los datos de interés a analizar.

Se van a importar los dos datasets disponibles: *train.csv* y *test.csv*. Las variables que tienen son las siguientes:

- **battery_power**: capacidad de la batería en mAh.
- **blue**: disponibilidad de bluetooth.
- **clock_speed**: frequencia de reloj del procesador.
- **dual_sim**: disponibilidad de Dual SIM.
- **fc**: megapíxeles de la cámara frontal.
- **four_g**: disponibilidad de 4G.
- **int_memory**: memoria interna en gigabytes.
- **m_dep**: profundidad del móvil en cm.
- **mobile_wt**: peso del móvil.
- **n_cores**: número de núcleos del procesador.
- **pc**: megapíxeles de la cámara principal.
- **px_height**: altura de resolución de pantalla.
- **px_width**: anchura de resolución de pantalla.
- **ram**: memoria RAM en megabytes.
- **sc_h**: altura de la pantalla en cm.
- **sc_w**: anchura de la pantalla en cm.
- **talk_time**: tiempo máximo de llamada.
- **three_g**: disponibilidad de 3G.
- **touch_screen**: disponibilidad de pantalla táctil.
- **wifi**: disponibilidad de wifi.
- **price_range**: rango de precio (0(bajo), 1(medio), 2(alto) and 3(muy alto)).

A priori, todas las variables son importantes para el estudio, por lo que van a ser incluidas.

```r
train<-read.csv(file = "../csv/train.csv")
test<-read.csv(file = "../csv/test.csv")

# informacion de las variables
str(train)
```

```
## 'data.frame':    2000 obs. of  21 variables:
##  $ battery_power: int  842 1021 563 615 1821 1859 1821 1954 1445 509 ...
##  $ blue         : int  0 1 1 1 1 0 0 0 1 1 ...
##  $ clock_speed  : num  2.2 0.5 0.5 2.5 1.2 0.5 1.7 0.5 0.5 0.6 ...
##  $ dual_sim     : int  0 1 1 0 0 1 0 1 0 1 ...
##  $ fc           : int  1 0 2 0 13 3 4 0 0 2 ...
##  $ four_g       : int  0 1 1 0 1 0 1 0 0 1 ...
##  $ int_memory   : int  7 53 41 10 44 22 10 24 53 9 ...
##  $ m_dep        : num  0.6 0.7 0.9 0.8 0.6 0.7 0.8 0.8 0.7 0.1 ...
##  $ mobile_wt    : int  188 136 145 131 141 164 139 187 174 93 ...
##  $ n_cores      : int  2 3 5 6 2 1 8 4 7 5 ...
##  $ pc           : int  2 6 6 9 14 7 10 0 14 15 ...
##  $ px_height    : int  20 905 1263 1216 1208 1004 381 512 386 1137 ...
##  $ px_width     : int  756 1988 1716 1786 1212 1654 1018 1149 836 1224 ...
##  $ ram          : int  2549 2631 2603 2769 1411 1067 3220 700 1099 513 ...
##  $ sc_h         : int  9 17 11 16 8 17 13 16 17 19 ...
##  $ sc_w         : int  7 3 2 8 2 1 8 3 1 10 ...
##  $ talk_time    : int  19 7 9 11 15 10 18 5 20 12 ...
##  $ three_g      : int  0 1 1 1 1 1 1 1 1 1 ...
##  $ touch_screen : int  0 1 1 0 1 0 0 1 0 0 ...
##  $ wifi         : int  1 0 0 0 0 0 1 1 0 0 ...
##  $ price_range  : int  1 2 2 2 1 1 3 0 0 0 ...
```

```r
summary(train)
```

```
##  battery_power        blue        clock_speed       dual_sim     
##  Min.   : 501.0   Min.   :0.000   Min.   :0.500   Min.   :0.0000  
##  1st Qu.: 851.8   1st Qu.:0.000   1st Qu.:0.700   1st Qu.:0.0000  
##  Median :1226.0   Median :0.000   Median :1.500   Median :1.0000  
##  Mean   :1238.5   Mean   :0.495   Mean   :1.522   Mean   :0.5095  
##  3rd Qu.:1615.2   3rd Qu.:1.000   3rd Qu.:2.200   3rd Qu.:1.0000  
##  Max.   :1998.0   Max.   :1.000   Max.   :3.000   Max.   :1.0000  
##        fc             four_g         int_memory        m_dep       
##  Min.   : 0.000   Min.   :0.0000   Min.   : 2.00   Min.   :0.1000  
##  1st Qu.: 1.000   1st Qu.:0.0000   1st Qu.:16.00   1st Qu.:0.2000  
##  Median : 3.000   Median :1.0000   Median :32.00   Median :0.5000  
##  Mean   : 4.309   Mean   :0.5215   Mean   :32.05   Mean   :0.5018  
##  3rd Qu.: 7.000   3rd Qu.:1.0000   3rd Qu.:48.00   3rd Qu.:0.8000  
##  Max.   :19.000   Max.   :1.0000   Max.   :64.00   Max.   :1.0000  
##    mobile_wt        n_cores           pc           px_height     
##  Min.   : 80.0   Min.   :1.000   Min.   : 0.000   Min.   :   0.0  
##  1st Qu.:109.0   1st Qu.:3.000   1st Qu.: 5.000   1st Qu.: 282.8  
##  Median :141.0   Median :4.000   Median :10.000   Median : 564.0  
##  Mean   :140.2   Mean   :4.521   Mean   : 9.916   Mean   : 645.1  
##  3rd Qu.:170.0   3rd Qu.:7.000   3rd Qu.:15.000   3rd Qu.: 947.2  
##  Max.   :200.0   Max.   :8.000   Max.   :20.000   Max.   :1960.0  
##     px_width           ram            sc_h            sc_w       
##  Min.   : 500.0   Min.   : 256   Min.   : 5.00   Min.   : 0.000  
##  1st Qu.: 874.8   1st Qu.:1208   1st Qu.: 9.00   1st Qu.: 2.000  
##  Median :1247.0   Median :2146   Median :12.00   Median : 5.000  
##  Mean   :1251.5   Mean   :2124   Mean   :12.31   Mean   : 5.767  
##  3rd Qu.:1633.0   3rd Qu.:3064   3rd Qu.:16.00   3rd Qu.: 9.000  
##  Max.   :1998.0   Max.   :3998   Max.   :19.00   Max.   :18.000  
##    talk_time        three_g        touch_screen        wifi      
##  Min.   : 2.00   Min.   :0.0000   Min.   :0.000   Min.   :0.000  
##  1st Qu.: 6.00   1st Qu.:1.0000   1st Qu.:0.000   1st Qu.:0.000  
##  Median :11.00   Median :1.0000   Median :1.000   Median :1.000  
##  Mean   :11.01   Mean   :0.7615   Mean   :0.503   Mean   :0.507  
##  3rd Qu.:16.00   3rd Qu.:1.0000   3rd Qu.:1.000   3rd Qu.:1.000  
##  Max.   :20.00   Max.   :1.0000   Max.   :1.000   Max.   :1.000  
##   price_range
##  Min.   :0.00
##  1st Qu.:0.75
##  Median :1.50
##  Mean   :1.50
##  3rd Qu.:2.25
##  Max.   :3.00
```

Limpieza de los datos.

Identificación y tratamiento de elementos vacíos.

```r
# valores nulos
colSums(is.na(train))
```

```
## battery_power          blue   clock_speed      dual_sim            fc
##             0             0             0             0             0
##        four_g    int_memory         m_dep     mobile_wt       n_cores
##             0             0             0             0             0
##            pc     px_height      px_width           ram          sc_h
##             0             0             0             0             0
##          sc_w     talk_time       three_g  touch_screen          wifi
##             0             0             0             0             0
##   price_range
##             0
```

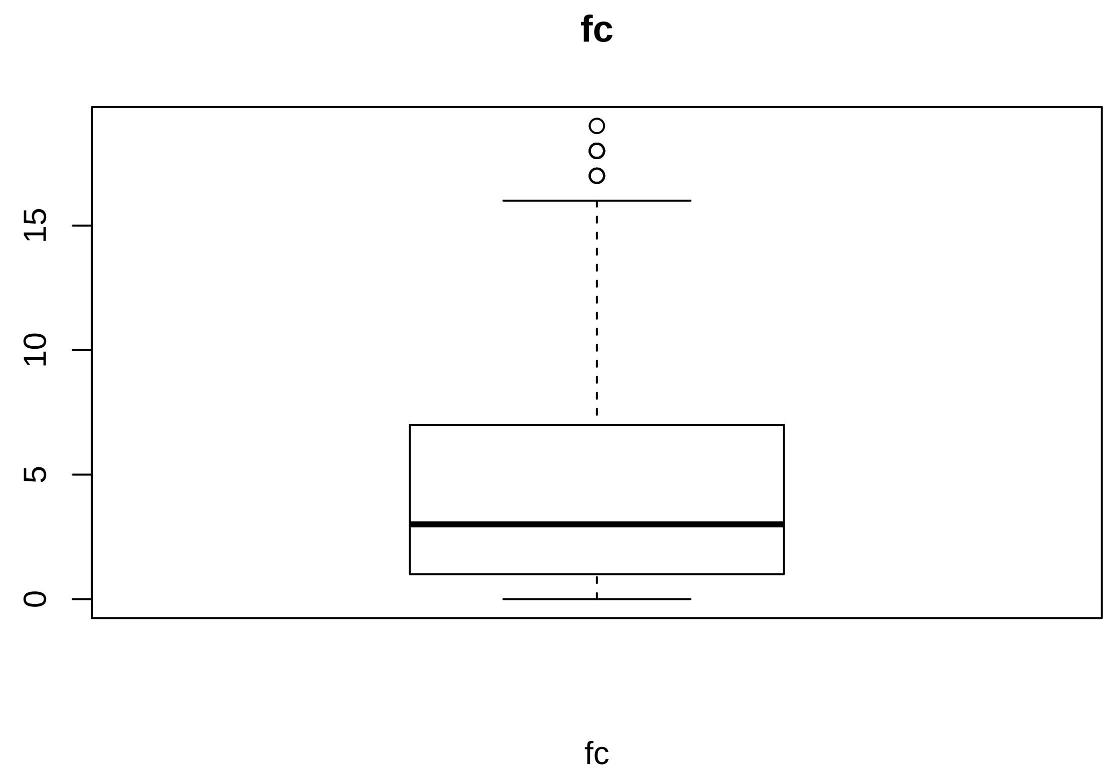
```r
colSums(is.na(test))
```

```
##            id battery_power          blue   clock_speed      dual_sim
##             0             0             0             0             0
##            fc        four_g    int_memory         m_dep     mobile_wt
##             0             0             0             0             0
##       n_cores            pc     px_height      px_width           ram
##             0             0             0             0             0
##          sc_h          sc_w     talk_time       three_g  touch_screen
##             0             0             0             0             0
##          wifi
##             0
```

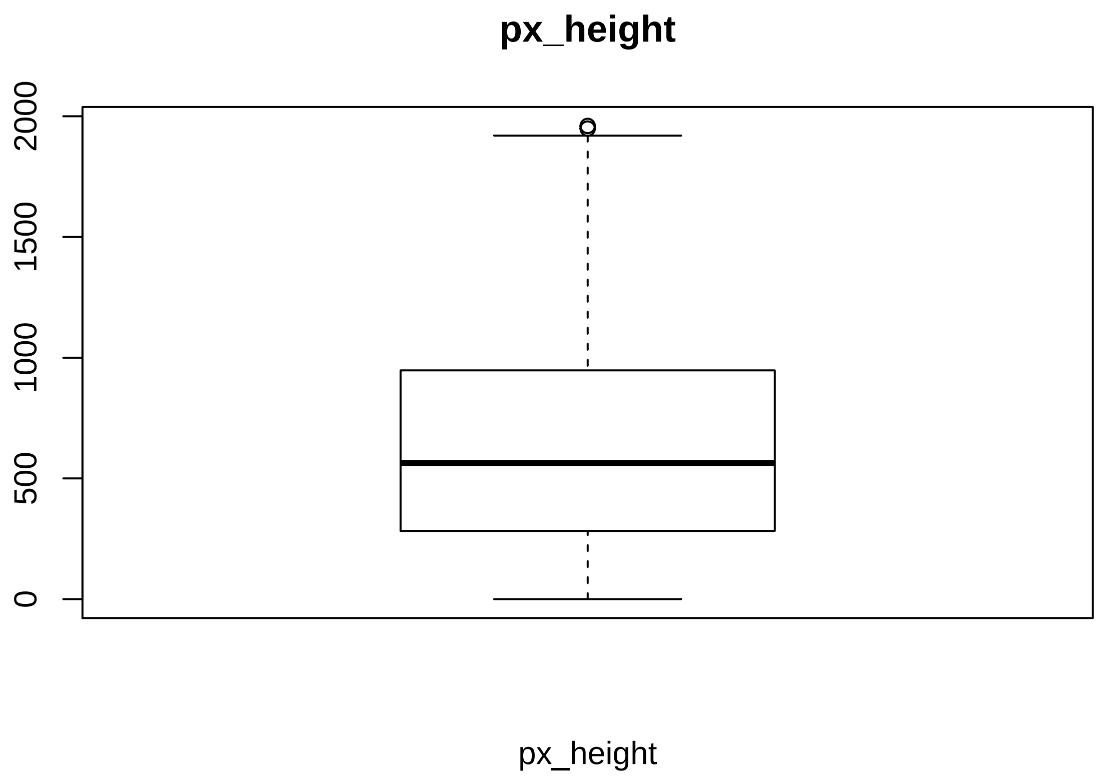
El dataset no contiene elementos vacíos.

Identificación y tratamiento de valores extremos.

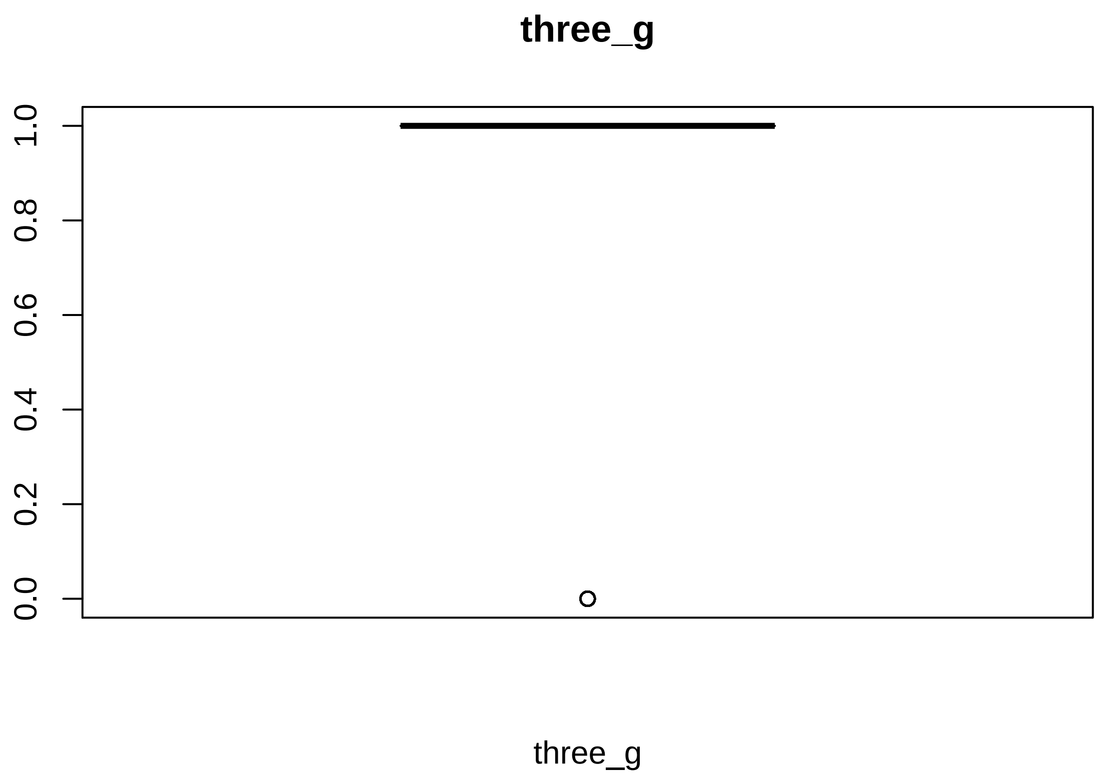
```r
for(i in colnames(train)){
  if(length(boxplot.stats(train[,i])$out) > 0)
  {
    boxplot(train[i], main = i, xlab = i)
    print(boxplot.stats(train[,i])$out)
  }
}
```



```
##  [1] 18 17 18 17 17 18 17 17 18 18 18 17 18 18 19 18 18 18
```



```
## [1] 1949 1960
```



```
##   [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [38] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [75] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [112] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [149] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [186] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [223] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [260] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [297] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [334] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [371] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [408] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [445] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Las variables *fc*, *px_height* y *three_g* tienen valores extremos. Sin embargo, todos ellos son valores reales (no indican lo contrario) por lo que no van a ser tratados.

Preprocesado.

Las variables van a ser tratadas para facilitar el análisis posterior.

Se van a discretizar las variables con pocas clases. Se van a formar grupos en el resto de las variables.

```r
train$ram = 2^trunc(log(train$ram,2))
test$ram = 2^trunc(log(test$ram,2))

# Discretizamos las variables con pocas clases
cols = c("blue", "dual_sim", "four_g", "n_cores", "ram", "three_g", "touch_screen",
         "wifi", "price_range")
for (i in cols){
  train[,i] = as.factor(train[,i])
}

cols = c("blue", "dual_sim", "four_g", "n_cores", "ram", "three_g", "touch_screen",
         "wifi")
for (i in cols){
  test[,i] = as.factor(test[,i])
}

# Discretizamos las demas variables en 5 grupos
# Generamos nuevas variables en lugar de remplazar las originales para poder usar ambas

rows = dim(train)[1]
rowsTest = dim(test)[1]


max = max(train$battery_power)
min = min(train$battery_power)
inter = (max - min) / 5
for (i in 1:rows){
  train$battery_power_group[i] = trunc((train$battery_power[i] - min) / inter)
}
train$battery_power_group = as.factor(train$battery_power_group)
levels(train$battery_power_group) = c("0", "1", "2", "3", "4", "4")
for (i in 1:rowsTest){
  test$battery_power_group[i] = trunc((test$battery_power[i] - min) / inter)
}
test$battery_power_group = as.factor(test$battery_power_group)
levels(test$battery_power_group) = c("0", "1", "2", "3", "4", "4")


max = max(train$clock_speed)
min = min(train$clock_speed)
inter = (max - min) / 5
for (i in 1:rows){
  train$clock_speed_group[i] = trunc((train$clock_speed[i] - min) / inter)
}
train$clock_speed_group = as.factor(train$clock_speed_group)
levels(train$clock_speed_group) = c("0", "1", "2", "3", "4", "4")
for (i in 1:rowsTest){
  test$clock_speed_group[i] = trunc((test$clock_speed[i] - min) / inter)
}
test$clock_speed_group = as.factor(test$clock_speed_group)
levels(test$clock_speed_group) = c("0", "1", "2", "3", "4", "4")
```

```r
max = max(train$fc)
min = min(train$fc)
inter = (max - min) / 5
for (i in 1:rows){
  train$fc_group[i] = trunc((train$fc[i] - min) / inter)
}
train$fc_group = as.factor(train$fc_group)
levels(train$fc_group) = c("0", "1", "2", "3", "4", "4")
for (i in 1:rowsTest){
  test$fc_group[i] = trunc((test$fc[i] - min) / inter)
}
test$fc_group = as.factor(test$fc_group)
levels(test$fc_group) = c("0", "1", "2", "3", "4", "4")


max = max(train$int_memory)
min = min(train$int_memory)
inter = (max - min) / 5
for (i in 1:rows){
  train$int_memory_group[i] = trunc((train$int_memory[i] - min) / inter)
}
train$int_memory_group = as.factor(train$int_memory_group)
levels(train$int_memory_group) = c("0", "1", "2", "3", "4", "4")
for (i in 1:rowsTest){
  test$int_memory_group[i] = trunc((test$int_memory[i] - min) / inter)
}
test$int_memory_group = as.factor(test$int_memory_group)
levels(test$int_memory_group) = c("0", "1", "2", "3", "4", "4")


max = max(train$m_dep)
min = min(train$m_dep)
inter = (max - min) / 5
for (i in 1:rows){
  train$m_dep_group[i] = trunc((train$m_dep[i] - min) / inter)
}
train$m_dep_group = as.factor(train$m_dep_group)
levels(train$m_dep_group) = c("0", "1", "2", "3", "4", "4")
for (i in 1:rowsTest){
  test$m_dep_group[i] = trunc((test$m_dep[i] - min) / inter)
}
test$m_dep_group = as.factor(test$m_dep_group)
levels(test$m_dep_group) = c("0", "1", "2", "3", "4", "4")


max = max(train$mobile_wt)
min = min(train$mobile_wt)
inter = (max - min) / 5
for (i in 1:rows){
  train$mobile_wt_group[i] = trunc((train$mobile_wt[i] - min) / inter)
}
train$mobile_wt_group = as.factor(train$mobile_wt_group)
levels(train$mobile_wt_group) = c("0", "1", "2", "3", "4", "4")
```

```
for (i in 1:rowsTest){
  test$mobile_wt_group[i] = trunc((test$mobile_wt[i] - min) / inter)
}
test$mobile_wt_group = as.factor(test$mobile_wt_group)
levels(test$mobile_wt_group) = c("0", "1", "2", "3", "4", "4")


max = max(train$pc)
min = min(train$pc)
inter = (max - min) / 5
for (i in 1:rows){
  train$pc_group[i] = trunc((train$pc[i] - min) / inter)
}
train$pc_group = as.factor(train$pc_group)
levels(train$pc_group) = c("0", "1", "2", "3", "4", "4")
for (i in 1:rowsTest){
  test$pc_group[i] = trunc((test$pc[i] - min) / inter)
}
test$pc_group = as.factor(test$pc_group)
levels(test$pc_group) = c("0", "1", "2", "3", "4", "4")


max = max(train$px_height)
min = min(train$px_height)
inter = (max - min) / 5
for (i in 1:rows){
  train$px_height_group[i] = trunc((train$px_height[i] - min) / inter)
}
train$px_height_group = as.factor(train$px_height_group)
levels(train$px_height_group) = c("0", "1", "2", "3", "4", "4")
for (i in 1:rowsTest){
  test$px_height_group[i] = trunc((test$px_height[i] - min) / inter)
}
test$px_height_group = as.factor(test$px_height_group)
levels(test$px_height_group) = c("0", "1", "2", "3", "4", "4")


max = max(train$px_width)
min = min(train$px_width)
inter = (max - min) / 5
for (i in 1:rows){
  train$px_width_group[i] = trunc((train$px_width[i] - min) / inter)
}
train$px_width_group = as.factor(train$px_width_group)
levels(train$px_width_group) = c("0", "1", "2", "3", "4", "4")
for (i in 1:rowsTest){
  test$px_width_group[i] = trunc((test$px_width[i] - min) / inter)
}
test$px_width_group = as.factor(test$px_width_group)
levels(test$px_width_group) = c("0", "1", "2", "3", "4", "4")


max = max(train$sc_h)
```

```r
min = min(train$sc_h)
inter = (max - min) / 5
for (i in 1:rows){
  train$sc_h_group[i] = trunc((train$sc_h[i] - min) / inter)
}
train$sc_h_group = as.factor(train$sc_h_group)
levels(train$sc_h_group) = c("0", "1", "2", "3", "4", "4")
for (i in 1:rowsTest){
  test$sc_h_group[i] = trunc((test$sc_h[i] - min) / inter)
}
test$sc_h_group = as.factor(test$sc_h_group)
levels(test$sc_h_group) = c("0", "1", "2", "3", "4", "4")


max = max(train$sc_w)
min = min(train$sc_w)
inter = (max - min) / 5
for (i in 1:rows){
  train$sc_w_group[i] = trunc((train$sc_w[i] - min) / inter)
}
train$sc_w_group = as.factor(train$sc_w_group)
levels(train$sc_w_group) = c("0", "1", "2", "3", "4", "4")
for (i in 1:rowsTest){
  test$sc_w_group[i] = trunc((test$sc_w[i] - min) / inter)
}
test$sc_w_group = as.factor(test$sc_w_group)
levels(test$sc_w_group) = c("0", "1", "2", "3", "4", "4")


max = max(train$talk_time)
min = min(train$talk_time)
inter = (max - min) / 5
for (i in 1:rows){
  train$talk_time_group[i] = trunc((train$talk_time[i] - min) / inter)
}
train$talk_time_group = as.factor(train$talk_time_group)
levels(train$talk_time_group) = c("0", "1", "2", "3", "4", "4")
for (i in 1:rowsTest){
  test$talk_time_group[i] = trunc((test$talk_time[i] - min) / inter)
}
test$talk_time_group = as.factor(test$talk_time_group)
levels(test$talk_time_group) = c("0", "1", "2", "3", "4", "4")


str(train)
```

```
## 'data.frame':    2000 obs. of  33 variables:
##  $ battery_power      : int  842 1021 563 615 1821 1859 1821 1954 1445 509 ...
##  $ blue               : Factor w/ 2 levels "0","1": 1 2 2 2 2 1 1 1 2 2 ...
##  $ clock_speed        : num  2.2 0.5 0.5 2.5 1.2 0.5 1.7 0.5 0.5 0.6 ...
##  $ dual_sim           : Factor w/ 2 levels "0","1": 1 2 2 1 1 2 1 2 1 2 ...
##  $ fc                 : int  1 0 2 0 13 3 4 0 0 2 ...
##  $ four_g             : Factor w/ 2 levels "0","1": 1 2 2 1 2 1 2 1 1 2 ...
##  $ int_memory         : int  7 53 41 10 44 22 10 24 53 9 ...
##  $ m_dep              : num  0.6 0.7 0.9 0.8 0.6 0.7 0.8 0.8 0.7 0.1 ...
##  $ mobile_wt          : int  188 136 145 131 141 164 139 187 174 93 ...
##  $ n_cores            : Factor w/ 8 levels "1","2","3","4",..: 2 3 5 6 2 1 8 4 7 5 ...
##  $ pc                 : int  2 6 6 9 14 7 10 0 14 15 ...
##  $ px_height          : int  20 905 1263 1216 1208 1004 381 512 386 1137 ...
##  $ px_width           : int  756 1988 1716 1786 1212 1654 1018 1149 836 1224 ...
##  $ ram                : Factor w/ 4 levels "256","512","1024",..: 4 4 4 4 3 3 4 2 3 2 ...
##  $ sc_h               : int  9 17 11 16 8 17 13 16 17 19 ...
##  $ sc_w               : int  7 3 2 8 2 1 8 3 1 10 ...
##  $ talk_time          : int  19 7 9 11 15 10 18 5 20 12 ...
##  $ three_g            : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 2 2 2 ...
##  $ touch_screen       : Factor w/ 2 levels "0","1": 1 2 2 1 2 1 1 2 1 1 ...
##  $ wifi               : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 2 2 1 1 ...
##  $ price_range        : Factor w/ 4 levels "0","1","2","3": 2 3 3 3 2 2 4 1 1 1 ...
##  $ battery_power_group: Factor w/ 5 levels "0","1","2","3",..: 2 2 1 1 5 5 5 5 4 1 ...
##  $ clock_speed_group  : Factor w/ 5 levels "0","1","2","3",..: 4 1 1 5 2 1 3 1 1 1 ...
##  $ fc_group           : Factor w/ 5 levels "0","1","2","3",..: 1 1 1 1 4 1 2 1 1 1 ...
##  $ int_memory_group   : Factor w/ 5 levels "0","1","2","3",..: 1 5 4 1 4 2 1 2 5 1 ...
##  $ m_dep_group        : Factor w/ 5 levels "0","1","2","3",..: 3 4 5 4 3 4 4 4 4 1 ...
##  $ mobile_wt_group    : Factor w/ 5 levels "0","1","2","3",..: 5 3 3 3 3 4 3 5 4 1 ...
##  $ pc_group           : Factor w/ 5 levels "0","1","2","3",..: 1 2 2 3 4 2 3 1 4 4 ...
##  $ px_height_group    : Factor w/ 5 levels "0","1","2","3",..: 1 3 4 4 4 3 1 2 1 3 ...
##  $ px_width_group     : Factor w/ 5 levels "0","1","2","3",..: 1 5 5 5 3 4 2 3 2 3 ...
##  $ sc_h_group         : Factor w/ 5 levels "0","1","2","3",..: 2 5 3 4 2 5 3 4 5 5 ...
##  $ sc_w_group         : Factor w/ 5 levels "0","1","2","3",..: 2 1 1 3 1 1 3 1 1 3 ...
##  $ talk_time_group    : Factor w/ 5 levels "0","1","2","3",..: 5 2 2 3 4 3 5 1 5 3 ...
```

Análisis de los datos

Selección de los grupos de datos que se quieren analizar/comparar (planificación de los análisis a aplicar).

Se van a generar varios dataframes útiles para los análisis que se realizarán a continuación:

- *train_set*: dataset con las variables seleccionadas para el estudio.
- *train_set_num*: copia de *train_set* pero con las variables de tipo numérico (necesario para ciertos test).
- *train_set_var*: dataset con las variables seleccionadas para el estudio sin la variable dependiente *price_range*.
- *train_set_var_num*: copia de *train_set_num* pero con las variables de tipo numérico.

```
vars1 = c("battery_power_group", "blue", "clock_speed_group", "dual_sim", "fc_group",
         "four_g", "int_memory_group", "m_dep_group", "mobile_wt_group", "n_cores",
         "pc_group", "px_height_group", "px_width_group", "ram", "sc_h_group", "sc_w_group",
         "talk_time_group", "three_g", "touch_screen", "wifi", "price_range")

vars2 = c("battery_power_group", "blue", "clock_speed_group", "dual_sim", "fc_group",
         "four_g", "int_memory_group", "m_dep_group", "mobile_wt_group", "n_cores",
         "pc_group", "px_height_group", "px_width_group", "ram", "sc_h_group", "sc_w_group",
         "talk_time_group", "three_g", "touch_screen", "wifi")
```

```r
train_set = train[vars1]

train_set_num = train_set
for (i in vars1){
  train_set_num[,i] = as.numeric(train_set[,i])
}

train_set_var = train_set[vars2]

train_set_var_num = train_set_var
for (i in vars2){
  train_set_var_num[,i] = as.numeric(train_set_var[,i])
}

str(train_set)
```

```
## 'data.frame':    2000 obs. of  21 variables:
##  $ battery_power_group: Factor w/ 5 levels "0","1","2","3",..: 2 2 1 1 5 5 5 5 4 1 ...
##  $ blue               : Factor w/ 2 levels "0","1": 1 2 2 2 2 1 1 1 2 2 ...
##  $ clock_speed_group  : Factor w/ 5 levels "0","1","2","3",..: 4 1 1 5 2 1 3 1 1 1 ...
##  $ dual_sim           : Factor w/ 2 levels "0","1": 1 2 2 1 1 2 1 2 1 2 ...
##  $ fc_group           : Factor w/ 5 levels "0","1","2","3",..: 1 1 1 1 4 1 2 1 1 1 ...
##  $ four_g             : Factor w/ 2 levels "0","1": 1 2 2 1 2 1 2 1 1 2 ...
##  $ int_memory_group   : Factor w/ 5 levels "0","1","2","3",..: 1 5 4 1 4 2 1 2 5 1 ...
##  $ m_dep_group        : Factor w/ 5 levels "0","1","2","3",..: 3 4 5 4 3 4 4 4 4 1 ...
##  $ mobile_wt_group    : Factor w/ 5 levels "0","1","2","3",..: 5 3 3 3 3 4 3 5 4 1 ...
##  $ n_cores            : Factor w/ 8 levels "1","2","3","4",..: 2 3 5 6 2 1 8 4 7 5 ...
##  $ pc_group           : Factor w/ 5 levels "0","1","2","3",..: 1 2 2 3 4 2 3 1 4 4 ...
##  $ px_height_group    : Factor w/ 5 levels "0","1","2","3",..: 1 3 4 4 4 3 1 2 1 3 ...
##  $ px_width_group     : Factor w/ 5 levels "0","1","2","3",..: 1 5 5 5 3 4 2 3 2 3 ...
##  $ ram                : Factor w/ 4 levels "256","512","1024",..: 4 4 4 4 3 3 4 2 3 2 ...
##  $ sc_h_group         : Factor w/ 5 levels "0","1","2","3",..: 2 5 3 4 2 5 3 4 5 5 ...
##  $ sc_w_group         : Factor w/ 5 levels "0","1","2","3",..: 2 1 1 3 1 1 3 1 1 3 ...
##  $ talk_time_group    : Factor w/ 5 levels "0","1","2","3",..: 5 2 2 3 4 3 5 1 5 3 ...
##  $ three_g            : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 2 2 2 ...
##  $ touch_screen       : Factor w/ 2 levels "0","1": 1 2 2 1 2 1 1 2 1 1 ...
##  $ wifi               : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 2 2 1 1 ...
##  $ price_range        : Factor w/ 4 levels "0","1","2","3": 2 3 3 3 2 2 4 1 1 1 ...
```

```r
str(train_set_num)
```

```
## 'data.frame':    2000 obs. of  21 variables:
##  $ battery_power_group: num  2 2 1 1 5 5 5 5 4 1 ...
##  $ blue               : num  1 2 2 2 2 1 1 1 2 2 ...
##  $ clock_speed_group  : num  4 1 1 5 2 1 3 1 1 1 ...
##  $ dual_sim           : num  1 2 2 1 1 2 1 2 1 2 ...
##  $ fc_group           : num  1 1 1 1 4 1 2 1 1 1 ...
##  $ four_g             : num  1 2 2 1 2 1 2 1 1 2 ...
##  $ int_memory_group   : num  1 5 4 1 4 2 1 2 5 1 ...
##  $ m_dep_group        : num  3 4 5 4 3 4 4 4 4 1 ...
##  $ mobile_wt_group    : num  5 3 3 3 3 4 3 5 4 1 ...
##  $ n_cores            : num  2 3 5 6 2 1 8 4 7 5 ...
##  $ pc_group           : num  1 2 2 3 4 2 3 1 4 4 ...
##  $ px_height_group    : num  1 3 4 4 4 3 1 2 1 3 ...
##  $ px_width_group     : num  1 5 5 5 3 4 2 3 2 3 ...
##  $ ram                : num  4 4 4 4 3 3 4 2 3 2 ...
##  $ sc_h_group         : num  2 5 3 4 2 5 3 4 5 5 ...
##  $ sc_w_group         : num  2 1 1 3 1 1 3 1 1 3 ...
##  $ talk_time_group    : num  5 2 2 3 4 3 5 1 5 3 ...
##  $ three_g            : num  1 2 2 2 2 2 2 2 2 2 ...
##  $ touch_screen       : num  1 2 2 1 2 1 1 2 1 1 ...
##  $ wifi               : num  2 1 1 1 1 1 2 2 1 1 ...
##  $ price_range        : num  2 3 3 3 2 2 4 1 1 1 ...
```

```r
str(train_set_var)
```

```
## 'data.frame':    2000 obs. of  20 variables:
##  $ battery_power_group: Factor w/ 5 levels "0","1","2","3",..: 2 2 1 1 5 5 5 5 4 1 ...
##  $ blue               : Factor w/ 2 levels "0","1": 1 2 2 2 2 1 1 1 2 2 ...
##  $ clock_speed_group  : Factor w/ 5 levels "0","1","2","3",..: 4 1 1 5 2 1 3 1 1 1 ...
##  $ dual_sim           : Factor w/ 2 levels "0","1": 1 2 2 1 1 2 1 2 1 2 ...
##  $ fc_group           : Factor w/ 5 levels "0","1","2","3",..: 1 1 1 1 4 1 2 1 1 1 ...
##  $ four_g             : Factor w/ 2 levels "0","1": 1 2 2 1 2 1 2 1 1 2 ...
##  $ int_memory_group   : Factor w/ 5 levels "0","1","2","3",..: 1 5 4 1 4 2 1 2 5 1 ...
##  $ m_dep_group        : Factor w/ 5 levels "0","1","2","3",..: 3 4 5 4 3 4 4 4 4 1 ...
##  $ mobile_wt_group    : Factor w/ 5 levels "0","1","2","3",..: 5 3 3 3 3 4 3 5 4 1 ...
##  $ n_cores            : Factor w/ 8 levels "1","2","3","4",..: 2 3 5 6 2 1 8 4 7 5 ...
##  $ pc_group           : Factor w/ 5 levels "0","1","2","3",..: 1 2 2 3 4 2 3 1 4 4 ...
##  $ px_height_group    : Factor w/ 5 levels "0","1","2","3",..: 1 3 4 4 4 3 1 2 1 3 ...
##  $ px_width_group     : Factor w/ 5 levels "0","1","2","3",..: 1 5 5 5 3 4 2 3 2 3 ...
##  $ ram                : Factor w/ 4 levels "256","512","1024",..: 4 4 4 4 3 3 4 2 3 2 ...
##  $ sc_h_group         : Factor w/ 5 levels "0","1","2","3",..: 2 5 3 4 2 5 3 4 5 5 ...
##  $ sc_w_group         : Factor w/ 5 levels "0","1","2","3",..: 2 1 1 3 1 1 3 1 1 3 ...
##  $ talk_time_group    : Factor w/ 5 levels "0","1","2","3",..: 5 2 2 3 4 3 5 1 5 3 ...
##  $ three_g            : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 2 2 2 ...
##  $ touch_screen       : Factor w/ 2 levels "0","1": 1 2 2 1 2 1 1 2 1 1 ...
##  $ wifi               : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 2 2 1 1 ...
```

```r
str(train_set_var_num)
```

```
## 'data.frame':    2000 obs. of  20 variables:
##  $ battery_power_group: num  2 2 1 1 5 5 5 5 4 1 ...
##  $ blue               : num  1 2 2 2 2 1 1 1 2 2 ...
##  $ clock_speed_group  : num  4 1 1 5 2 1 3 1 1 1 ...
##  $ dual_sim           : num  1 2 2 1 1 2 1 2 1 2 ...
##  $ fc_group           : num  1 1 1 1 4 1 2 1 1 1 ...
##  $ four_g             : num  1 2 2 1 2 1 2 1 1 2 ...
##  $ int_memory_group   : num  1 5 4 1 4 2 1 2 5 1 ...
##  $ m_dep_group        : num  3 4 5 4 3 4 4 4 4 1 ...
##  $ mobile_wt_group    : num  5 3 3 3 3 4 3 5 4 1 ...
##  $ n_cores            : num  2 3 5 6 2 1 8 4 7 5 ...
##  $ pc_group           : num  1 2 2 3 4 2 3 1 4 4 ...
##  $ px_height_group    : num  1 3 4 4 4 3 1 2 1 3 ...
##  $ px_width_group     : num  1 5 5 5 3 4 2 3 2 3 ...
##  $ ram                : num  4 4 4 4 3 3 4 2 3 2 ...
##  $ sc_h_group         : num  2 5 3 4 2 5 3 4 5 5 ...
##  $ sc_w_group         : num  2 1 1 3 1 1 3 1 1 3 ...
##  $ talk_time_group    : num  5 2 2 3 4 3 5 1 5 3 ...
##  $ three_g            : num  1 2 2 2 2 2 2 2 2 2 ...
##  $ touch_screen       : num  1 2 2 1 2 1 1 2 1 1 ...
##  $ wifi               : num  2 1 1 1 1 1 2 2 1 1 ...
```

Comprobación de la normalidad.

Se van a realizar los tests de Kolmogorov-Smirnov y de Shapiro-Wilk para comprobar la normlaidad de las variables. Si el p-valor es menor al nivel de significancia, $\alpha = 0,05$ por defecto, se considera que los datos no cuentan con una distribución normal.

```
nor_table=matrix(nc=2, nr=0)
colnames(nor_table)=c("shapiro.test" ," ks.test")

for (i in colnames(train_set_var_num)){
  s_test = shapiro.test(train_set_var_num[,i])
  ks_test = ks.test(train_set_var_num[,i], pnorm, mean(train_set_var_num[,i]), sd(train_set_var_num[,i]

  row=matrix(nc=2, nr=1)
  row[1][1]= s_test$p.value
  row[2][1]= ks_test$p.value

  nor_table=rbind(nor_table, row)
  rownames(nor_table)[nrow(nor_table)]=i
}

print(nor_table)
```

```
##                     shapiro.test  ks.test
## battery_power_group 2.741765e-36        0
## blue                5.074999e-54        0
## clock_speed_group   4.777715e-40        0
## dual_sim            5.023046e-54        0
## fc_group            1.232272e-47        0
## four_g              4.736917e-54        0
## int_memory_group    2.351863e-36        0
## m_dep_group         8.483054e-37        0
## mobile_wt_group     3.314660e-36        0
## n_cores             2.753812e-30        0
## pc_group            1.201248e-36        0
## px_height_group     1.030644e-39        0
## px_width_group      9.256638e-36        0
## ram                 2.681737e-47        0
## sc_h_group          8.353563e-36        0
## sc_w_group          8.291552e-41        0
## talk_time_group     7.211151e-37        0
## three_g             1.418438e-58        0
## touch_screen        5.087821e-54        0
## wifi                5.055827e-54        0
```

Los datos no siguen una distribución normal.

Pruebas estadísticas.

¿Cuáles son las variables que determinan en mayor medida el precio?

Se va a utilizar la correlación de Spearman para ver el grado de dependencia de la variable *price_range* con el resto de variables.

```r
# tabla de correlacion

corr=matrix(nc=2, nr=0)
colnames(corr)=c("estimate" ," p-value")

for(i in colnames(train_set_num)){
   spearman_test=cor.test(train_set_num[,i],
                          train_set_num$price_range,
                          method="spearman")

   row=matrix(nc=2, nr=1)
   row[1][1]= spearman_test$estimate
   row[2][1]= spearman_test$p.value

   corr=rbind(corr, row)
   rownames(corr)[nrow(corr)]=i

}
print(corr)
```

```
##                        estimate      p-value
## battery_power_group  0.199312887 2.290324e-19
## blue                 0.020572854 3.577985e-01
## clock_speed_group   -0.001392615 9.503711e-01
## dual_sim             0.017444479 4.355602e-01
## fc_group             0.018755035 4.018622e-01
## four_g               0.014771711 5.091037e-01
## int_memory_group     0.038704817 8.354128e-02
## m_dep_group         -0.003817480 8.645251e-01
## mobile_wt_group     -0.028765542 1.984794e-01
## n_cores              0.004651407 8.353162e-01
## pc_group             0.030196613 1.770490e-01
## px_height_group      0.129202401 6.678521e-09
## px_width_group       0.159219042 7.964104e-13
## ram                  0.847236953 0.000000e+00
## sc_h_group           0.020421391 3.613495e-01
## sc_w_group           0.029277814 1.906002e-01
## talk_time_group      0.020228461 3.659046e-01
## three_g              0.023611217 2.912366e-01
## touch_screen        -0.030411072 1.739918e-01
## wifi                 0.018784812 4.011152e-01
## price_range          1.000000000 0.000000e+00
```

Se observa una correlación importante entre la variable *ram* y *price_range*.

Hipotesis. Relación entre poca RAM y precio bajo.

```
# Hipotesis nula: las variables son independientes.
# Hipotesis alternativa: las variables no son independientes

train_set_num = transform(train_set_num, lowRAM= ifelse(ram<3, TRUE, FALSE))
train_set_num = transform(train_set_num, lowPrice= ifelse(price_range<3, TRUE, FALSE))

tablaContingencia = table(train_set_num$lowRAM, train_set_num$lowPrice)

tablaContingenciaMarg = addmargins(tablaContingencia)
tablaContingenciaMarg
```

```
##
##          FALSE TRUE  Sum
##   FALSE  1000  594 1594
##   TRUE      0  406  406
##   Sum    1000 1000 2000
```

```
test = chisq.test(tablaContingenciaMarg, correct=FALSE)

# Resultado con la funcion chisq.test
test$p.value
```

```
## [1] 6.175908e-109
```

Como el p-valor es muy pequeño (casi 0) se tienen evidencias para decir que no son independientes.

Modelo de regresión lineal.

```
mobileTrain_set = train_set
col_set = colnames(mobileTrain_set)
for (i in col_set){
  mobileTrain_set[,i] = as.numeric(mobileTrain_set[,i])
}

regresion1 = lm(price_range ~ ram, data = mobileTrain_set)

regresion2 = lm(price_range ~ ram + battery_power_group, data = mobileTrain_set)

regresion3 = lm(price_range ~ ram + battery_power_group + px_height_group + four_g,
                data = mobileTrain_set)

regresion4 = lm(price_range ~ ram + battery_power_group + px_height_group + four_g
                + n_cores, data = mobileTrain_set)

# Tabla resultado
coef=matrix(c( 1, summary(regresion1)$r.squared,
               2, summary(regresion2)$r.squared,
               3, summary(regresion3)$r.squared,
               4, summary(regresion4)$r.squared),
               ncol=2, byrow=TRUE)

colnames(coef)=c("Modelo" ,"R^2")
coef
```

```
##      Modelo       R^2
## [1,]      1 0.6521680
## [2,]      2 0.6938815
## [3,]      3 0.7189329
## [4,]      4 0.7190155
```

Conclusiones.

A partir de un conjunto de datos sobre la venta de móviles se han podido dar respuesta a preguntas como cuáles son las variables que más influyen en el precio final, o para establecer un modelo que permita prececir el precio de un nuevo teléfono.

El procesado previo de los datos es muy importante para conseguir que el análisis sea eficiente y eficaz.

Equipo

Contribuciones	Firma
Investigación previa	PML
Redacción de las respuestas	PML
Desarrollo código	PML

Referencias

[1] Dataset *Mobile Price Classification* (https://www.kaggle.com/iabhishekofficial/mobile-price-classification)