Proyecto 2. Entrega 5. Regresión logística

Pablo Daniel Barillas Moreno, Carné No. 22193
Mathew Cordero Aquino, Carné No. 22982

2025-03-14

Enlace al Repositorio del proyecto 2 - Entrega 5 de minería de datos del Grupo #1

Repositorio en GitHub

0. Descargue los conjuntos de datos.

Para este punto, ya se ha realizado el proceso para descargar del sitio web: House Prices - Advanced Regression Techniques, la data de entrenamiento y la data de prueba, ambos extraídos desde la carpeta "house_prices_data/" en data frames llamados train_data (data de entrenamiento) y test_data (data de prueba), sin convertir automáticamente las variables categóricas en factores (stringsAsFactors = FALSE). Luego, se realiza una inspección inicial de train_data mediante tres funciones: head(train_data), que muestra las primeras filas del dataset; str(train_data), que despliega la estructura del data frame, incluyendo el tipo de cada variable; y summary(train_data), que proporciona un resumen estadístico de las variables numéricas y una descripción general de las categóricas.

```
train_data <- read.csv("train_set.csv", stringsAsFactors = FALSE)
test_data <- read.csv("test_set.csv", stringsAsFactors = FALSE)

head(train_data)   # Muestra las primeras filas
```

```
##   Id MSSubClass MSZoning LotFrontage LotArea Street LotShape LandContour
## 1  3         60       RL          68   11250   Pave      IR1         Lvl
## 2  5         60       RL          84   14260   Pave      IR1         Lvl
## 3  7         20       RL          75   10084   Pave      Reg         Lvl
## 4  8         60       RL          69   10382   Pave      IR1         Lvl
## 5 10        190       RL          50    7420   Pave      Reg         Lvl
## 6 13         20       RL          69   12968   Pave      IR2         Lvl
##   Utilities LotConfig LandSlope Neighborhood Condition1 Condition2 BldgType
## 1    AllPub    Inside       Gtl      CollgCr       Norm       Norm     1Fam
## 2    AllPub       FR2       Gtl      NoRidge       Norm       Norm     1Fam
## 3    AllPub    Inside       Gtl      Somerst       Norm       Norm     1Fam
## 4    AllPub    Corner       Gtl       NWAmes       PosN       Norm     1Fam
## 5    AllPub    Corner       Gtl      BrkSide     Artery     Artery   2fmCon
## 6    AllPub    Inside       Gtl       Sawyer       Norm       Norm     1Fam
##   HouseStyle OverallQual OverallCond YearBuilt YearRemodAdd RoofStyle RoofMatl
## 1     2Story           7           5      2001         2002     Gable  CompShg
## 2     2Story           8           5      2000         2000     Gable  CompShg
## 3     1Story           8           5      2004         2005     Gable  CompShg
## 4     2Story           7           6      1973         1973     Gable  CompShg
```

```
## 5     1.5Unf           5           6      1939         1950     Gable  CompShg
## 6     1Story           5           6      1962         1962       Hip  CompShg
##   Exterior1st Exterior2nd MasVnrType MasVnrArea ExterQual ExterCond Foundation
## 1     VinylSd     VinylSd    BrkFace        162        Gd        TA      PConc
## 2     VinylSd     VinylSd    BrkFace        350        Gd        TA      PConc
## 3     VinylSd     VinylSd      Stone        186        Gd        TA      PConc
## 4     HdBoard     HdBoard      Stone        240        TA        TA     CBlock
## 5     MetalSd     MetalSd       None          0        TA        TA     BrkTil
## 6     HdBoard     Plywood       None          0        TA        TA     CBlock
##   BsmtQual BsmtCond BsmtExposure BsmtFinType1 BsmtFinSF1 BsmtFinType2
## 1       Gd       TA           Mn          GLQ        486          Unf
## 2       Gd       TA           Av          GLQ        655          Unf
## 3       Ex       TA           Av          GLQ       1369          Unf
## 4       Gd       TA           Mn          ALQ        859          BLQ
## 5       TA       TA           No          GLQ        851          Unf
## 6       TA       TA           No          ALQ        737          Unf
##   BsmtFinSF2 BsmtUnfSF TotalBsmtSF Heating HeatingQC CentralAir Electrical
## 1          0       434         920    GasA        Ex          Y      SBrkr
## 2          0       490        1145    GasA        Ex          Y      SBrkr
## 3          0       317        1686    GasA        Ex          Y      SBrkr
## 4         32       216        1107    GasA        Ex          Y      SBrkr
## 5          0       140         991    GasA        Ex          Y      SBrkr
## 6          0       175         912    GasA        TA          Y      SBrkr
##   X1stFlrSF X2ndFlrSF LowQualFinSF GrLivArea BsmtFullBath BsmtHalfBath FullBath
## 1       920       866            0      1786            1            0        2
## 2      1145      1053            0      2198            1            0        2
## 3      1694         0            0      1694            1            0        2
## 4      1107       983            0      2090            1            0        2
## 5      1077         0            0      1077            1            0        1
## 6       912         0            0       912            1            0        1
##   HalfBath BedroomAbvGr KitchenAbvGr KitchenQual TotRmsAbvGrd Functional
## 1        1            3            1          Gd            6        Typ
## 2        1            4            1          Gd            9        Typ
## 3        0            3            1          Gd            7        Typ
## 4        1            3            1          TA            7        Typ
## 5        0            2            2          TA            5        Typ
## 6        0            2            1          TA            4        Typ
##   Fireplaces FireplaceQu GarageType GarageYrBlt GarageFinish GarageCars
## 1          1          TA     Attchd        2001          RFn          2
## 2          1          TA     Attchd        2000          RFn          3
## 3          1          Gd     Attchd        2004          RFn          2
## 4          2          TA     Attchd        1973          RFn          2
## 5          2          TA     Attchd        1939          RFn          1
## 6          0        None     Detchd        1962          Unf          1
##   GarageArea GarageQual GarageCond PavedDrive WoodDeckSF OpenPorchSF
## 1        608         TA         TA          Y          0          42
## 2        836         TA         TA          Y        192          84
## 3        636         TA         TA          Y        255          57
## 4        484         TA         TA          Y        235         204
## 5        205         Gd         TA          Y          0           4
## 6        352         TA         TA          Y        140           0
##   EnclosedPorch X3SsnPorch ScreenPorch PoolArea MiscVal MoSold YrSold SaleType
## 1             0          0           0        0       0      9   2008       WD
## 2             0          0           0        0       0     12   2008       WD
```

```
## 3             0          0           0        0       0      8   2007       WD
## 4           228          0           0        0     350     11   2009       WD
## 5             0          0           0        0       0      1   2008       WD
## 6             0          0         176        0       0      9   2008       WD
##   SaleCondition SalePrice LogSalePrice QualityGroup SizeGroup Cluster Age
## 1        Normal    223500     12.31717        Media   Mediana       2   7
## 2        Normal    250000     12.42922         Alta    Grande       1   8
## 3        Normal    307000     12.63460         Alta   Mediana       1   3
## 4        Normal    200000     12.20607        Media    Grande       2  36
## 5        Normal    118000     11.67844        Media   Mediana       3  69
## 6        Normal    144000     11.87757        Media   Pequeña       3  46
##   Qual_LivArea SalePriceCat
## 1        12502         cara
## 2        17584         cara
## 3        13552         cara
## 4        14630         cara
## 5         5385       barata
## 6         4560        media
```

```
str(train_data)    # Muestra la estructura del dataset
```

```
## 'data.frame':    937 obs. of  84 variables:
##  $ Id           : int  3 5 7 8 10 13 15 18 19 20 ...
##  $ MSSubClass   : int  60 60 20 60 190 20 20 90 20 20 ...
##  $ MSZoning     : chr  "RL" "RL" "RL" "RL" ...
##  $ LotFrontage  : int  68 84 75 69 50 69 69 72 66 70 ...
##  $ LotArea      : int  11250 14260 10084 10382 7420 12968 10920 10791 13695 7560 ...
##  $ Street       : chr  "Pave" "Pave" "Pave" "Pave" ...
##  $ LotShape     : chr  "IR1" "IR1" "Reg" "IR1" ...
##  $ LandContour  : chr  "Lvl" "Lvl" "Lvl" "Lvl" ...
##  $ Utilities    : chr  "AllPub" "AllPub" "AllPub" "AllPub" ...
##  $ LotConfig    : chr  "Inside" "FR2" "Inside" "Corner" ...
##  $ LandSlope    : chr  "Gtl" "Gtl" "Gtl" "Gtl" ...
##  $ Neighborhood : chr  "CollgCr" "NoRidge" "Somerst" "NWAmes" ...
##  $ Condition1   : chr  "Norm" "Norm" "Norm" "PosN" ...
##  $ Condition2   : chr  "Norm" "Norm" "Norm" "Norm" ...
##  $ BldgType     : chr  "1Fam" "1Fam" "1Fam" "1Fam" ...
##  $ HouseStyle   : chr  "2Story" "2Story" "1Story" "2Story" ...
##  $ OverallQual  : int  7 8 8 7 5 5 6 4 5 5 ...
##  $ OverallCond  : int  5 5 5 6 6 6 5 5 5 6 ...
##  $ YearBuilt    : int  2001 2000 2004 1973 1939 1962 1960 1967 2004 1958 ...
##  $ YearRemodAdd : int  2002 2000 2005 1973 1950 1962 1960 1967 2004 1965 ...
##  $ RoofStyle    : chr  "Gable" "Gable" "Gable" "Gable" ...
##  $ RoofMatl     : chr  "CompShg" "CompShg" "CompShg" "CompShg" ...
##  $ Exterior1st  : chr  "VinylSd" "VinylSd" "VinylSd" "HdBoard" ...
##  $ Exterior2nd  : chr  "VinylSd" "VinylSd" "VinylSd" "HdBoard" ...
##  $ MasVnrType   : chr  "BrkFace" "BrkFace" "Stone" "Stone" ...
##  $ MasVnrArea   : int  162 350 186 240 0 0 212 0 0 0 ...
##  $ ExterQual    : chr  "Gd" "Gd" "Gd" "TA" ...
##  $ ExterCond    : chr  "TA" "TA" "TA" "TA" ...
##  $ Foundation   : chr  "PConc" "PConc" "PConc" "CBlock" ...
##  $ BsmtQual     : chr  "Gd" "Gd" "Ex" "Gd" ...
##  $ BsmtCond     : chr  "TA" "TA" "TA" "TA" ...
```

```
##  $ BsmtExposure : chr  "Mn" "Av" "Av" "Mn" ...
##  $ BsmtFinType1 : chr  "GLQ" "GLQ" "GLQ" "ALQ" ...
##  $ BsmtFinSF1   : int  486 655 1369 859 851 737 733 0 646 504 ...
##  $ BsmtFinType2 : chr  "Unf" "Unf" "Unf" "BLQ" ...
##  $ BsmtFinSF2   : int  0 0 0 32 0 0 0 0 0 0 ...
##  $ BsmtUnfSF    : int  434 490 317 216 140 175 520 0 468 525 ...
##  $ TotalBsmtSF  : int  920 1145 1686 1107 991 912 1253 0 1114 1029 ...
##  $ Heating      : chr  "GasA" "GasA" "GasA" "GasA" ...
##  $ HeatingQC    : chr  "Ex" "Ex" "Ex" "Ex" ...
##  $ CentralAir   : chr  "Y" "Y" "Y" "Y" ...
##  $ Electrical   : chr  "SBrkr" "SBrkr" "SBrkr" "SBrkr" ...
##  $ X1stFlrSF    : int  920 1145 1694 1107 1077 912 1253 1296 1114 1339 ...
##  $ X2ndFlrSF    : int  866 1053 0 983 0 0 0 0 0 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1786 2198 1694 2090 1077 912 1253 1296 1114 1339 ...
##  $ BsmtFullBath : int  1 1 1 1 1 1 1 0 1 0 ...
##  $ BsmtHalfBath : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  2 2 2 2 1 1 1 2 1 1 ...
##  $ HalfBath     : int  1 1 0 1 0 0 1 0 1 0 ...
##  $ BedroomAbvGr : int  3 4 3 3 2 2 2 2 3 3 ...
##  $ KitchenAbvGr : int  1 1 1 1 2 1 1 2 1 1 ...
##  $ KitchenQual  : chr  "Gd" "Gd" "Gd" "TA" ...
##  $ TotRmsAbvGrd : int  6 9 7 7 5 4 5 6 6 6 ...
##  $ Functional   : chr  "Typ" "Typ" "Typ" "Typ" ...
##  $ Fireplaces   : int  1 1 1 2 2 0 1 0 0 0 ...
##  $ FireplaceQu  : chr  "TA" "TA" "Gd" "TA" ...
##  $ GarageType   : chr  "Attchd" "Attchd" "Attchd" "Attchd" ...
##  $ GarageYrBlt  : int  2001 2000 2004 1973 1939 1962 1960 1967 2004 1958 ...
##  $ GarageFinish : chr  "RFn" "RFn" "RFn" "RFn" ...
##  $ GarageCars   : int  2 3 2 2 1 1 1 2 2 1 ...
##  $ GarageArea   : int  608 836 636 484 205 352 352 516 576 294 ...
##  $ GarageQual   : chr  "TA" "TA" "TA" "TA" ...
##  $ GarageCond   : chr  "TA" "TA" "TA" "TA" ...
##  $ PavedDrive   : chr  "Y" "Y" "Y" "Y" ...
##  $ WoodDeckSF   : int  0 192 255 235 0 140 0 0 0 0 ...
##  $ OpenPorchSF  : int  42 84 57 204 4 0 213 0 102 0 ...
##  $ EnclosedPorch: int  0 0 0 228 0 0 176 0 0 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ScreenPorch  : int  0 0 0 0 0 176 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ MiscVal      : int  0 0 0 350 0 0 0 500 0 0 ...
##  $ MoSold       : int  9 12 8 11 1 9 5 10 6 5 ...
##  $ YrSold       : int  2008 2008 2007 2009 2008 2008 2008 2006 2008 2009 ...
##  $ SaleType     : chr  "WD" "WD" "WD" "WD" ...
##  $ SaleCondition: chr  "Normal" "Normal" "Normal" "Normal" ...
##  $ SalePrice    : num  223500 250000 307000 200000 118000 ...
##  $ LogSalePrice : num  12.3 12.4 12.6 12.2 11.7 ...
##  $ QualityGroup : chr  "Media" "Alta" "Alta" "Media" ...
##  $ SizeGroup    : chr  "Mediana" "Grande" "Mediana" "Grande" ...
##  $ Cluster      : int  2 1 1 2 3 3 3 3 2 3 ...
##  $ Age          : int  7 8 3 36 69 46 48 39 4 51 ...
##  $ Qual_LivArea : int  12502 17584 13552 14630 5385 4560 7518 5184 5570 6695 ...
##  $ SalePriceCat : chr  "cara" "cara" "cara" "cara" ...
```

```r
summary(train_data) # Resumen estadístico
```

```
##        Id           MSSubClass       MSZoning          LotFrontage
##  Min.   :   3.0   Min.   : 20.00   Length:937         Min.   : 21.00
##  1st Qu.: 364.0   1st Qu.: 20.00   Class :character   1st Qu.: 60.00
##  Median : 728.0   Median : 50.00   Mode  :character   Median : 69.00
##  Mean   : 729.1   Mean   : 55.67                      Mean   : 69.88
##  3rd Qu.:1094.0   3rd Qu.: 70.00                      3rd Qu.: 79.00
##  Max.   :1459.0   Max.   :190.00                      Max.   :313.00
##
##     LotArea          Street            LotShape         LandContour
##  Min.   :  1477   Length:937         Length:937         Length:937
##  1st Qu.:  7596   Class :character   Class :character   Class :character
##  Median :  9405   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 10234
##  3rd Qu.: 11643
##  Max.   :115149
##
##   Utilities          LotConfig          LandSlope         Neighborhood
##  Length:937         Length:937         Length:937         Length:937
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   Condition1         Condition2          BldgType          HouseStyle
##  Length:937         Length:937         Length:937         Length:937
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   OverallQual      OverallCond      YearBuilt     YearRemodAdd
##  Min.   : 1.000   Min.   :1.000   Min.   :1875   Min.   :1950
##  1st Qu.: 5.000   1st Qu.:5.000   1st Qu.:1953   1st Qu.:1967
##  Median : 6.000   Median :5.000   Median :1973   Median :1994
##  Mean   : 6.079   Mean   :5.606   Mean   :1971   Mean   :1985
##  3rd Qu.: 7.000   3rd Qu.:6.000   3rd Qu.:2000   3rd Qu.:2003
##  Max.   :10.000   Max.   :9.000   Max.   :2009   Max.   :2010
##
##   RoofStyle           RoofMatl         Exterior1st        Exterior2nd
##  Length:937         Length:937         Length:937         Length:937
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   MasVnrType          MasVnrArea       ExterQual          ExterCond
##  Length:937         Min.   :   0.00   Length:937         Length:937
##  Class :character   1st Qu.:   0.00   Class :character   Class :character
```

```
##  Mode  :character   Median :   0.00   Mode  :character   Mode  :character
##                     Mean   :  99.48
##                     3rd Qu.: 157.75
##                     Max.   :1600.00
##                     NA's   :7
##   Foundation          BsmtQual           BsmtCond         BsmtExposure
##  Length:937         Length:937         Length:937         Length:937
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  BsmtFinType1         BsmtFinSF1   BsmtFinType2         BsmtFinSF2
##  Length:937         Min.   :   0   Length:937         Min.   :   0.0
##  Class :character   1st Qu.:   0   Class :character   1st Qu.:   0.0
##  Mode  :character   Median : 374   Mode  :character   Median :   0.0
##                     Mean   : 441                      Mean   :  50.6
##                     3rd Qu.: 713                      3rd Qu.:   0.0
##                     Max.   :5644                      Max.   :1474.0
##
##    BsmtUnfSF       TotalBsmtSF     Heating           HeatingQC
##  Min.   :   0.0   Min.   :   0   Length:937         Length:937
##  1st Qu.: 218.0   1st Qu.: 798   Class :character   Class :character
##  Median : 479.0   Median : 990   Mode  :character   Mode  :character
##  Mean   : 570.1   Mean   :1062
##  3rd Qu.: 813.0   3rd Qu.:1278
##  Max.   :2336.0   Max.   :6110
##
##   CentralAir         Electrical          X1stFlrSF      X2ndFlrSF
##  Length:937         Length:937         Min.   : 438   Min.   :   0.0
##  Class :character   Class :character   1st Qu.: 894   1st Qu.:   0.0
##  Mode  :character   Mode  :character   Median :1085   Median :   0.0
##                                        Mean   :1169   Mean   : 341.9
##                                        3rd Qu.:1390   3rd Qu.: 728.0
##                                        Max.   :4692   Max.   :2065.0
##
##   LowQualFinSF       GrLivArea     BsmtFullBath     BsmtHalfBath
##  Min.   :  0.000   Min.   : 438   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:  0.000   1st Qu.:1124   1st Qu.:0.0000   1st Qu.:0.00000
##  Median :  0.000   Median :1471   Median :0.0000   Median :0.00000
##  Mean   :  3.289   Mean   :1515   Mean   :0.4312   Mean   :0.05229
##  3rd Qu.:  0.000   3rd Qu.:1795   3rd Qu.:1.0000   3rd Qu.:0.00000
##  Max.   :572.000   Max.   :5642   Max.   :3.0000   Max.   :2.00000
##
##     FullBath        HalfBath       BedroomAbvGr    KitchenAbvGr
##  Min.   :0.000   Min.   :0.0000   Min.   :0.000   Min.   :0.000
##  1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:2.000   1st Qu.:1.000
##  Median :2.000   Median :0.0000   Median :3.000   Median :1.000
##  Mean   :1.577   Mean   :0.3831   Mean   :2.876   Mean   :1.052
##  3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.:3.000   3rd Qu.:1.000
##  Max.   :3.000   Max.   :2.0000   Max.   :6.000   Max.   :3.000
##
##  KitchenQual         TotRmsAbvGrd    Functional          Fireplaces
```

```
##  Length:937         Min.   : 3.00   Length:937         Min.   :0.0000
##  Class :character   1st Qu.: 5.00   Class :character   1st Qu.:0.0000
##  Mode  :character   Median : 6.00   Mode  :character   Median :1.0000
##                     Mean   : 6.53                      Mean   :0.6009
##                     3rd Qu.: 7.00                      3rd Qu.:1.0000
##                     Max.   :12.00                      Max.   :3.0000
##
##  FireplaceQu         GarageType         GarageYrBlt   GarageFinish
##  Length:937         Length:937         Min.   :1900   Length:937
##  Class :character   Class :character   1st Qu.:1962   Class :character
##  Mode  :character   Mode  :character   Median :1979   Mode  :character
##                                        Mean   :1979
##                                        3rd Qu.:2001
##                                        Max.   :2010
##                                        NA's   :54
##    GarageCars      GarageArea      GarageQual         GarageCond
##  Min.   :0.000   Min.   :   0.0   Length:937         Length:937
##  1st Qu.:1.000   1st Qu.: 318.0   Class :character   Class :character
##  Median :2.000   Median : 478.0   Mode  :character   Mode  :character
##  Mean   :1.756   Mean   : 470.9
##  3rd Qu.:2.000   3rd Qu.: 576.0
##  Max.   :4.000   Max.   :1418.0
##
##   PavedDrive          WoodDeckSF      OpenPorchSF    EnclosedPorch
##  Length:937         Min.   :  0.00   Min.   :  0.0   Min.   :  0.00
##  Class :character   1st Qu.:  0.00   1st Qu.:  0.0   1st Qu.:  0.00
##  Mode  :character   Median :  0.00   Median : 25.0   Median :  0.00
##                     Mean   : 93.47   Mean   : 46.6   Mean   : 23.55
##                     3rd Qu.:168.00   3rd Qu.: 69.0   3rd Qu.:  0.00
##                     Max.   :857.00   Max.   :502.0   Max.   :386.00
##
##    X3SsnPorch       ScreenPorch        PoolArea          MiscVal
##  Min.   :  0.000   Min.   :  0.00   Min.   :  0.000   Min.   :    0.00
##  1st Qu.:  0.000   1st Qu.:  0.00   1st Qu.:  0.000   1st Qu.:    0.00
##  Median :  0.000   Median :  0.00   Median :  0.000   Median :    0.00
##  Mean   :  2.995   Mean   : 15.94   Mean   :  3.138   Mean   :   35.15
##  3rd Qu.:  0.000   3rd Qu.:  0.00   3rd Qu.:  0.000   3rd Qu.:    0.00
##  Max.   :407.000   Max.   :440.00   Max.   :738.000   Max.   :15500.00
##
##      MoSold           YrSold       SaleType         SaleCondition
##  Min.   : 1.000   Min.   :2006   Length:937         Length:937
##  1st Qu.: 5.000   1st Qu.:2007   Class :character   Class :character
##  Median : 6.000   Median :2008   Mode  :character   Mode  :character
##  Mean   : 6.383   Mean   :2008
##  3rd Qu.: 8.000   3rd Qu.:2009
##  Max.   :12.000   Max.   :2010
##
##    SalePrice       LogSalePrice   QualityGroup        SizeGroup
##  Min.   : 35311   Min.   :10.47   Length:937         Length:937
##  1st Qu.:130000   1st Qu.:11.78   Class :character   Class :character
##  Median :163000   Median :12.00   Mode  :character   Mode  :character
##  Mean   :180334   Mean   :12.02
##  3rd Qu.:214000   3rd Qu.:12.27
##  Max.   :745000   Max.   :13.52
```

```
##
##     Cluster           Age          Qual_LivArea   SalePriceCat
##  Min.   :1.000   Min.   :  0.00   Min.   :  876   Length:937
##  1st Qu.:2.000   1st Qu.:  8.00   1st Qu.: 5720   Class :character
##  Median :2.000   Median : 35.00   Median : 8806   Mode  :character
##  Mean   :2.187   Mean   : 36.78   Mean   : 9649
##  3rd Qu.:3.000   3rd Qu.: 55.00   3rd Qu.:12327
##  Max.   :3.000   Max.   :135.00   Max.   :56420
##
```

1. Cree una variable dicotómica por cada una de las categorías de la variable respuesta categórica que creó en hojas anteriores. Debería tener 3 variables dicotómicas (valores 0 y 1) una que diga si la vivienda es cara o no, media o no, económica o no.

Codificación de variables dicotómicas

En entregas anteriores, se transformó la variable `SalePrice` en una variable categórica que clasifica los precios de las viviendas en tres niveles: `barata`, `media` y `cara`. Esta categorización permite abordar el problema de predicción desde una perspectiva de clasificación. Para adaptar estos datos a modelos de regresión logística binaria, es necesario generar nuevas variables dicotómicas que indiquen si una observación pertenece o no a cada una de estas categorías.

A continuación, se entrena un modelo de regresión logística binaria para predecir si una vivienda pertenece a la categoría `cara`. Se parte de los archivos `train_set.csv` y `test_set.csv`, asegurando que la variable categórica `SalePriceCat` esté correctamente definida, y que la variable binaria `es_cara` sea coherente.

```r
# Librerías necesarias
library(dplyr)
library(caret)

# Cargar datos regenerados
train <- read.csv("train_set.csv")
test  <- read.csv("test_set.csv")

# Crear variables dicotómicas a partir de SalePriceCat
train <- train %>%
  mutate(
    es_barata = ifelse(SalePriceCat == "barata", 1, 0),
    es_media  = ifelse(SalePriceCat == "media", 1, 0),
    es_cara   = ifelse(SalePriceCat == "cara", 1, 0)
  )

test <- test %>%
  mutate(
    es_barata = ifelse(SalePriceCat == "barata", 1, 0),
    es_media  = ifelse(SalePriceCat == "media", 1, 0),
    es_cara   = ifelse(SalePriceCat == "cara", 1, 0)
  )

# Verificación rápida
cat("Frecuencia de clases en train:\n")
```

```
## Frecuencia de clases en train:
```

```r
print(table(train$SalePriceCat))
```

```
##
## barata   cara  media
##    313    312    312
```

```r
cat("\nFrecuencia binaria es_cara:\n")
```

```
##
## Frecuencia binaria es_cara:
```

```r
print(table(train$es_cara))
```

```
##
##   0   1
## 625 312
```

```r
# Variable de respuesta binaria como factor
train$es_cara <- factor(ifelse(train$es_cara == 1, "caro", "no_caro"))

# Modelo simplificado con variables relevantes
set.seed(123)
ctrl <- trainControl(
  method = "cv",
  number = 10,
  sampling = "up",
  savePredictions = TRUE
)

modelo_rl_final <- train(
  es_cara ~ OverallQual + GrLivArea + GarageCars + TotalBsmtSF + YearBuilt,
  data = train,
  method = "glm",
  family = "binomial",
  trControl = ctrl,
  metric = "Accuracy",
  preProcess = c("center", "scale", "medianImpute")
)

# Resultados del modelo
print(modelo_rl_final)
```

```
## Generalized Linear Model
##
## 937 samples
##   5 predictor
##   2 classes: 'caro', 'no_caro'
##
## Pre-processing: centered (5), scaled (5), median imputation (5)
## Resampling: Cross-Validated (10 fold)
```

```
## Summary of sample sizes: 842, 844, 843, 844, 844, 844, ...
## Addtional sampling using up-sampling prior to pre-processing
##
## Resampling results:
##
##   Accuracy   Kappa
##   0.8890726  0.7607836
```

```
summary(modelo_rl_final$finalModel)
```

```
##
## Call:
## NULL
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.1088     0.1042   1.045  0.29615
## OverallQual  -1.5572     0.2030  -7.671 1.70e-14 ***
## GrLivArea    -1.9016     0.1865 -10.195  < 2e-16 ***
## GarageCars   -0.6063     0.2164  -2.801  0.00509 **
## TotalBsmtSF  -0.7461     0.1372  -5.440 5.34e-08 ***
## YearBuilt    -0.6773     0.1445  -4.688 2.76e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1732.9  on 1249  degrees of freedom
## Residual deviance:  662.3  on 1244  degrees of freedom
## AIC: 674.3
##
## Number of Fisher Scoring iterations: 7
```

Este modelo es ahora completamente funcional, con una variable respuesta válida (`es_cara`) y predictores seleccionados por su relevancia. Se entrena utilizando validación cruzada de 10 folds con balanceo (upsampling) para evitar el desbalance de clases. Los resultados mostrarán coeficientes significativos y métricas de desempeño útiles como `Accuracy`.

Serie 1: Creación de variables dicotómicas y modelo para viviendas caras

En esta serie se llevó a cabo la transformación de la variable categórica `SalePriceCat` en **tres variables dicotómicas**:

- `es_barata`: vale 1 si la vivienda es barata, 0 en caso contrario.
- `es_media`: vale 1 si la vivienda es media, 0 en caso contrario.
- `es_cara`: vale 1 si la vivienda es cara, 0 en caso contrario.

Estas variables permiten modelar cada categoría de forma binaria, lo cual es útil para ajustar modelos independientes con regresión logística.

Frecuencia de clases en el conjunto de entrenamiento

```r
table(train$SalePriceCat)
```

Categoría	Frecuencia
barata	313
media	312
cara	312

La distribución de clases es **equilibrada**, lo cual es ideal para entrenar modelos binarios sin sesgo.

Frecuencia de la variable `es_cara`

```r
table(train$es_cara)
```

Valor	Significado	Frecuencia
0	no es vivienda cara	625
1	es vivienda cara	312

Esto confirma que la variable binaria `es_cara` fue correctamente generada.

Modelo de regresión logística para predecir viviendas caras

Se construyó un modelo de regresión logística para predecir si una vivienda es `cara` utilizando 5 variables predictoras relevantes. El modelo se entrenó con validación cruzada de 10 folds, preprocesamiento (centrado, escalado e imputación), y balanceo por **upsampling**.

Resultados del modelo

- **Accuracy promedio**: `0.889`
- **Kappa**: `0.761`
  → Indican un buen rendimiento predictivo y acuerdo entre clases.

Coeficientes estimados

Variable	Coeficiente	Significancia	Interpretación
OverallQual	-1.56	***	A mayor calidad general, menor prob. de ser cara
GrLivArea	-1.90	***	Mayor área habitable, menor probabilidad
GarageCars	-0.61	**	Más garaje, menor probabilidad
TotalBsmtSF	-0.75	***	Sótano más grande, menor probabilidad
YearBuilt	-0.68	***	Vivienda más reciente, menor probabilidad

> Todos los predictores son **estadísticamente significativos** (`p < 0.01`), lo que valida su uso en el modelo.

Ajuste del modelo

- **Null deviance**: 1732.9
- **Residual deviance**: 662.3

- **AIC**: 674.3

Estos valores indican que el modelo explica de forma sustancial la variabilidad de la variable respuesta `es_cara`.

Conclusión:

La Serie 1 cumplió con éxito su objetivo de crear variables dicotómicas a partir de la categorización del precio, y de entrenar un modelo de regresión logística válido, interpretable y con buen rendimiento para identificar viviendas caras.

2. Use los mismos conjuntos de entrenamiento y prueba que utilizó en las hojas anteriores.

Reutilización de los conjuntos de entrenamiento y prueba

Para mantener la coherencia metodológica y asegurar una comparación justa entre los modelos de clasificación construidos en las diferentes entregas del proyecto, se ha reutilizado la misma partición de los datos en los conjuntos de entrenamiento y prueba definida previamente.

Estos conjuntos (`train_set.csv` y `test_set.csv`) fueron generados al inicio del proyecto a partir del conjunto de datos original de Kaggle. La partición se realizó utilizando una semilla aleatoria fija para garantizar reproducibilidad en todas las etapas del análisis.

A continuación, se presenta el código para cargar los archivos correspondientes:

```
# Cargar conjuntos de datos ya particionados en entregas anteriores
train <- read.csv("train_set.csv")
test  <- read.csv("test_set.csv")

# Verificar tamaño de los conjuntos
cat("Observaciones en el conjunto de entrenamiento:", nrow(train), "\n")
```

```
## Observaciones en el conjunto de entrenamiento: 937
```

```
cat("Observaciones en el conjunto de prueba:", nrow(test), "\n")
```

```
## Observaciones en el conjunto de prueba: 232
```

```
# Validar estructura y columna categórica
str(train$SalePrice)
```

```
##  num [1:937] 223500 250000 307000 200000 118000 ...
```

```
table(train$SalePrice)
```

```
##
##  35311  37900  40000  52000  55993  60000  61000  64500  66500  67000  68500
##      1      1      1      1      1      3      1      1      1      1      1
##  72500  73000  75000  75500  76000  78000  79000  79500  79900  80000  81000
##      1      1      1      1      1      1      2      1      1      3      3
##  82000  82500  83000  83500  84000  84500  84900  85000  85400  85500  86000
##      1      1      1      1      1      2      1      3      1      1      2
```

```
##  87000  87500  88000  89000  89471  89500  90000  90350  91000  91300  91500
##      3      1      2      1      1      1      3      1      2      1      2
##  92900  93000  93500  94000  94500  95000  96500  98000  98300  99500  1e+05
##      1      3      2      1      1      2      1      1      1      1      9
## 102000 103000 103200 104900 105000 105500 106000 106250 107000 107400 107500
##      1      1      1      1      5      1      1      1      1      1      2
## 108000 108480 108500 108959 109008 109500 109900 110000 110500 111000 111250
##      4      1      1      1      1      2      2      7      1      1      1
## 112000 112500 113000 114504 115000 116000 116500 117000 117500 118000 118400
##      5      1      3      1      7      2      1      3      2      4      1
## 118500 118858 118964 119000 119200 119500 119750 119900 120000 120500 121600
##      3      1      1      3      1      2      1      1      4      3      1
## 122000 122500 122900 123000 123500 123600 124000 124500 124900 125000 126000
##      3      1      1      3      1      1      3      2      1      9      3
## 126500 127000 127500 128000 128200 128500 128900 129000 129500 129900 130000
##      1      6      5      5      1      2      1      3      2      3      6
## 130250 130500 131000 131400 131500 132000 132500 133000 133700 133900 134000
##      1      3      2      1      1      3      6      5      1      2      3
## 134432 134500 134800 134900 135000 135500 135750 135900 135960 136000 136500
##      1      1      1      1      8      2      1      1      1      3      5
## 136900 137000 137450 137500 137900 138500 138800 139000 139400 139500 139600
##      1      2      1      4      1      2      1     10      1      1      1
## 139900 139950 140000 141000 141500 142000 142125 142500 142600 142953 143000
##      1      1     16      4      1      3      1      3      1      1      5
## 143500 143900 144000 144152 144900 145000 145250 145500 145900 146000 146500
##      2      1      7      1      1      5      1      1      1      2      1
## 147000 147400 148000 148500 149000 149350 149500 149700 149900 150000 150500
##      5      1      5      1      2      1      1      1      2      3      1
## 150900 151000 151500 152000 153000 153337 153500 153575 153900 154000 154300
##      1      2      1      4      1      1      2      1      2      2      1
## 154500 155000 156000 156500 157000 157500 157900 158000 158500 158900 159000
##      1     11      3      1      3      1      2      5      1      1      3
## 159434 159500 159895 159950 160000 160200 161500 161750 162000 162900 163000
##      1      2      1      1      7      1      1      1      2      2      2
## 163500 163900 164000 164500 164990 165000 165150 165400 165500 165600 166000
##      2      1      1      2      1      5      1      1      2      1      2
## 167000 167500 168000 168500 169000 169900 169990 170000 171000 171750 172400
##      3      2      1      1      2      1      1      6      2      1      1
## 172500 173000 173500 173900 174000 175000 175500 175900 176000 176485 176500
##      5      5      1      1      4      5      3      2      4      1      2
## 177000 177500 178000 178400 178900 179000 179200 179400 179500 179540 179600
##      2      2      6      1      1      3      2      1      1      1      1
## 179900 180000 180500 181000 181134 182000 182900 183200 184000 184100 184750
##      5      6      3      3      1      1      1      1      3      1      1
## 185000 185750 185850 186500 187000 187100 187500 188000 188500 188700 189000
##      4      1      1      1      1      1      4      2      1      1      5
## 190000 191000 192000 192500 193000 193500 193879 194000 194500 195000 196500
##     10      4      5      1      3      1      1      1      2      2      1
## 197000 197500 197900 198500 198900 199900  2e+05 200100 200624 201000 201800
##      2      2      3      1      1      1      6      1      1      3      1
## 202500 202900 203000 204000 204750 204900 205000 205950 206000 206300 206900
##      2      1      1      1      1      1      6      1      1      1      1
## 207000 207500 208300 208900 209500 210000 211000 212000 213000 213250 213490
##      2      3      1      2      1      3      2      2      2      1      1
```

```
## 213500 214000 214500 214900 215000 215200 216000 216500 216837 217000 217500
##      1      5      1      1      6      1      1      1      1      2      1
## 218000 219210 219500 220000 221000 221500 222000 222500 223000 223500 224000
##      1      1      2      2      1      1      2      1      1      2      1
## 224900 225000 226000 226700 227000 227680 227875 228000 228500 229000 229456
##      1      3      3      1      2      1      1      1      2      1      1
## 230000 230500 231500 232000 233000 233170 233230 234000 235000 235128 236000
##      5      1      1      2      1      1      1      2      3      1      1
## 236500 237000 237500 238000 239000 239686 240000 241000 241500 242000 243000
##      2      2      1      1      4      1      3      1      2      1      1
## 244000 244600 245000 245350 245500 246578 248900 249700 250000 250580 251000
##      2      1      1      1      1      1      1      1      6      1      1
## 252678 253000 253293 254900 255000 255500 255900 256000 256300 257000 257500
##      1      1      1      1      1      1      1      1      1      1      1
## 258000 259000 259500 260000 262000 262500 263000 265900 265979 266000 266500
##      1      1      1      2      1      1      1      1      1      1      1
## 267000 268000 269790 270000 271000 271900 274000 274725 274970 275000 278000
##      1      1      1      3      2      1      1      1      1      3      1
## 280000 281213 282922 283463 284000 286000 287000 289000 290000 294000 295000
##      4      1      1      1      1      1      2      1      2      1      1
## 297000 299800 301000 301500 302000 303477 305000 307000 309000 310000 311500
##      1      1      1      1      1      1      1      1      1      1      1
## 312500 313000 314813 315000 315500 315750 316600 317000 319000 320000 325000
##      1      1      1      3      1      1      1      1      1      3      3
## 325300 325624 326000 328000 328900 335000 337000 340000 341000 342643 348000
##      1      1      1      1      1      1      1      1      1      1      1
## 350000 360000 361919 367294 372500 374000 377500 378500 385000 386250 392000
##      2      1      1      1      1      1      1      1      2      1      1
## 392500 394617 395000 403000 412500 415298 424870 426000 430000 438780 446261
##      1      1      1      1      1      1      1      1      1      1      1
## 465000 466500 475000 501837 538000 555000 556581 611657 625000 745000
##      1      1      1      1      1      1      1      1      1      1
```

El uso de estos mismos conjuntos permite evaluar el desempeño de los modelos en condiciones equivalentes, facilitando una comparación válida entre algoritmos como Árboles de Decisión, Random Forest, Naive Bayes, KNN y Regresión Logística, todos entrenados y probados con estos mismos datos.

Serie 2: Análisis de la variable `SalePrice` en el conjunto de entrenamiento

Para tener una mejor comprensión del comportamiento del precio de venta de las viviendas (`SalePrice`), se realizó un análisis exploratorio sobre el conjunto de entrenamiento.

Dimensión del conjunto

- Observaciones en el conjunto de entrenamiento: **937**
- Observaciones en el conjunto de prueba: **232**

Vista preliminar de los precios

Los valores de `SalePrice` varían ampliamente en el conjunto de entrenamiento, con precios desde los **35,311** hasta los **745,000**, lo cual refleja una gran heterogeneidad en las propiedades evaluadas. Este comportamiento refuerza la necesidad de una transformación o categorización para modelar esta variable de manera más efectiva.

Frecuencia de precios

Se utilizó la función `table()` sobre `SalePrice` para obtener las frecuencias absolutas de cada precio observado. A partir de estos resultados, se destacan los siguientes hallazgos:

- Hay múltiples precios únicos que solo aparecen **una vez**, indicando una gran dispersión.
- Algunos valores de precio aparecen con mayor frecuencia, como:
  - **100000**: aparece **9 veces**
  - **125000**: aparece **9 veces**
  - **135000**: aparece **8 veces**
  - **140000**: aparece **16 veces**
  - **150000**: aparece **3 veces**
  - **160000**: aparece **7 veces**

Estos picos de frecuencia podrían deberse a precios de lista comunes, umbrales de negociación o valores de referencia dentro del mercado.

Observación general

El análisis muestra que, aunque `SalePrice` es una variable numérica continua, en la práctica tiende a agruparse alrededor de valores "redondeados", lo que sugiere la viabilidad de agruparla en categorías como **barata**, **media** y **cara**, para facilitar el modelado como problema de clasificación. Esta categorización fue efectuada en entregas anteriores a través de cuantiles.

3. Elabore un modelo de regresión logística para conocer si una vivienda es cara o no, utilizando el conjunto de entrenamiento y explique los resultados a los que llega. El experimento debe ser reproducible por lo que debe fijar que los conjuntos de entrenamiento y prueba sean los mismos siempre que se ejecute el código. Use validación cruzada.

Modelo de regresión logística para predecir si una vivienda es cara

Para conocer qué factores influyen en que una vivienda sea clasificada como `cara`, se construye un modelo de regresión logística binaria utilizando como variable respuesta la columna `es_cara`, previamente generada a partir de la variable categórica `SalePriceCat`.

El modelo se entrena sobre el conjunto de datos `train_set.csv`, previamente particionado de forma estratificada y reproducible. Se utiliza validación cruzada de 10 particiones (`k = 10`) y se aplica balanceo mediante **upsampling**, dada la menor proporción de casos `caro` respecto a `no_caro`.

Además, se aplica preprocesamiento: centrado, escalado e imputación de valores faltantes por la mediana.

```r
# Librerías necesarias
library(dplyr)
library(caret)

# Fijar semilla para garantizar reproducibilidad
set.seed(123)

# Cargar datos regenerados
train <- read.csv("train_set.csv")

# Crear variable dicotómica correctamente desde SalePriceCat
train <- train %>%
  mutate(
    es_cara = ifelse(SalePriceCat == "cara", 1, 0)
  )

# Convertir a factor binario (para caret)
train$es_cara <- factor(ifelse(train$es_cara == 1, "caro", "no_caro"))
```

```r
# Configuración de validación cruzada estratificada con balanceo
ctrl <- trainControl(
  method = "cv",
  number = 10,
  sampling = "up",  # balanceo por upsampling
  savePredictions = TRUE
)

# Entrenamiento del modelo con variables relevantes
modelo_rl <- train(
  es_cara ~ OverallQual + GrLivArea + GarageCars + TotalBsmtSF + YearBuilt,
  data = train,
  method = "glm",
  family = "binomial",
  trControl = ctrl,
  metric = "Accuracy",
  preProcess = c("center", "scale", "medianImpute")
)

# Resultados del modelo
print(modelo_rl)
```

```
## Generalized Linear Model
##
## 937 samples
##   5 predictor
##   2 classes: 'caro', 'no_caro'
##
## Pre-processing: centered (5), scaled (5), median imputation (5)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 842, 844, 843, 844, 844, 844, ...
## Addtional sampling using up-sampling prior to pre-processing
##
## Resampling results:
##
##   Accuracy   Kappa
##   0.8890726  0.7607836
```

```r
summary(modelo_rl$finalModel)
```

```
##
## Call:
## NULL
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.1088     0.1042   1.045  0.29615
## OverallQual  -1.5572     0.2030  -7.671 1.70e-14 ***
## GrLivArea    -1.9016     0.1865 -10.195  < 2e-16 ***
## GarageCars   -0.6063     0.2164  -2.801  0.00509 **
## TotalBsmtSF  -0.7461     0.1372  -5.440 5.34e-08 ***
## YearBuilt    -0.6773     0.1445  -4.688 2.76e-06 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1732.9  on 1249  degrees of freedom
## Residual deviance:  662.3  on 1244  degrees of freedom
## AIC: 674.3
##
## Number of Fisher Scoring iterations: 7
```

El modelo entrenado permite identificar la probabilidad de que una vivienda sea clasificada como cara en función de variables clave como:

- **OverallQual**: calidad general de los materiales y acabados.
- **GrLivArea**: superficie habitable sobre el nivel del suelo.
- **GarageCars**: capacidad del garaje (número de vehículos).
- **TotalBsmtSF**: superficie total del sótano.
- **YearBuilt**: año de construcción.

El resumen de los coeficientes del modelo (`summary(modelo_rl$finalModel)`) permite observar cuáles variables son estadísticamente significativas (valores p bajos) y si su efecto es positivo o negativo sobre la probabilidad de ser una vivienda cara. Esto se analizará con mayor detalle en la siguiente sección.

Modelo simplificado de regresión logística para predecir si una vivienda es cara

Dado que los modelos anteriores presentaron problemas de sobreajuste y predicción trivial, en esta sección se entrena un modelo simplificado utilizando únicamente variables altamente relacionadas con el precio de la vivienda. Esto reduce la complejidad, mejora la interpretabilidad y evita la multicolinealidad.

Resultados del modelo

El modelo de regresión logística entrenado para predecir si una vivienda es cara (`es_cara`) obtuvo resultados satisfactorios tanto en desempeño predictivo como en interpretación estadística:

- **Accuracy promedio**: `0.889`
- **Kappa**: `0.76`
  Esto indica que el modelo clasifica correctamente el 88.9% de las observaciones, y tiene un acuerdo sustancial entre las clases (`caro` / `no_caro`) más allá del azar.

Análisis de coeficientes

El resumen del modelo (`summary(modelo_rl$finalModel)`) muestra los siguientes resultados:

Variable	Coeficiente	Significancia (p-valor)	Interpretación
OverallQual	-1.557	***	A mayor calidad general, **menor** prob. de ser cara
GrLivArea	-1.902	***	A mayor superficie habitable, **menor** probabilidad
GarageCars	-0.606	**	Más espacio de garaje, **menor** prob. de ser cara
TotalBsmtSF	-0.746	***	Sótano más grande → menor probabilidad de ser cara
YearBuilt	-0.677	***	Viviendas más nuevas tienden a ser menos caras

Nota: Estos coeficientes negativos indican que estas variables están inversamente asociadas con la probabilidad de que una vivienda sea clasificada como `cara` dentro del contexto del conjunto de datos, posiblemente porque el precio ya fue categorizado y otras variables lo explican mejor.

Métricas del modelo

- **Null deviance**: 1732.9 → devianza del modelo sin predictores.
- **Residual deviance**: 662.3 → mejora sustancial al incluir predictores.
- **AIC**: 674.3 → buena medida de ajuste; menor es mejor.

En conclusión, el modelo tiene un **buen desempeño predictivo**, y las variables seleccionadas son estadísticamente significativas. Esto valida su uso para identificar patrones asociados a viviendas clasificadas como caras en el conjunto de datos.

Serie 3: Categorización de `SalePrice` y representación gráfica

Dada la gran dispersión en los valores de la variable `SalePrice`, se procedió a **categorizarlos** en tres niveles: **barata**, **media** y **cara**, utilizando los terciles (cuantiles 1/3 y 2/3) como puntos de corte. Esta transformación permite tratar el problema como una tarea de **clasificación multiclase**, en lugar de regresión continua.

Categorización por cuantiles

```
# Cargar librerías
library(dplyr)
library(ggplot2)

# Cargar datos base si no están en memoria
train <- read.csv("train_set.csv")

# Crear variable categórica SalePriceCat basada en terciles
quantiles <- quantile(train$SalePrice, probs = c(1/3, 2/3))
train$SalePriceCat <- cut(
  train$SalePrice,
  breaks = c(-Inf, quantiles[1], quantiles[2], Inf),
  labels = c("barata", "media", "cara")
)

# Verificar distribución
cat("Distribución de categorías:\n")
```

```
## Distribución de categorías:
```

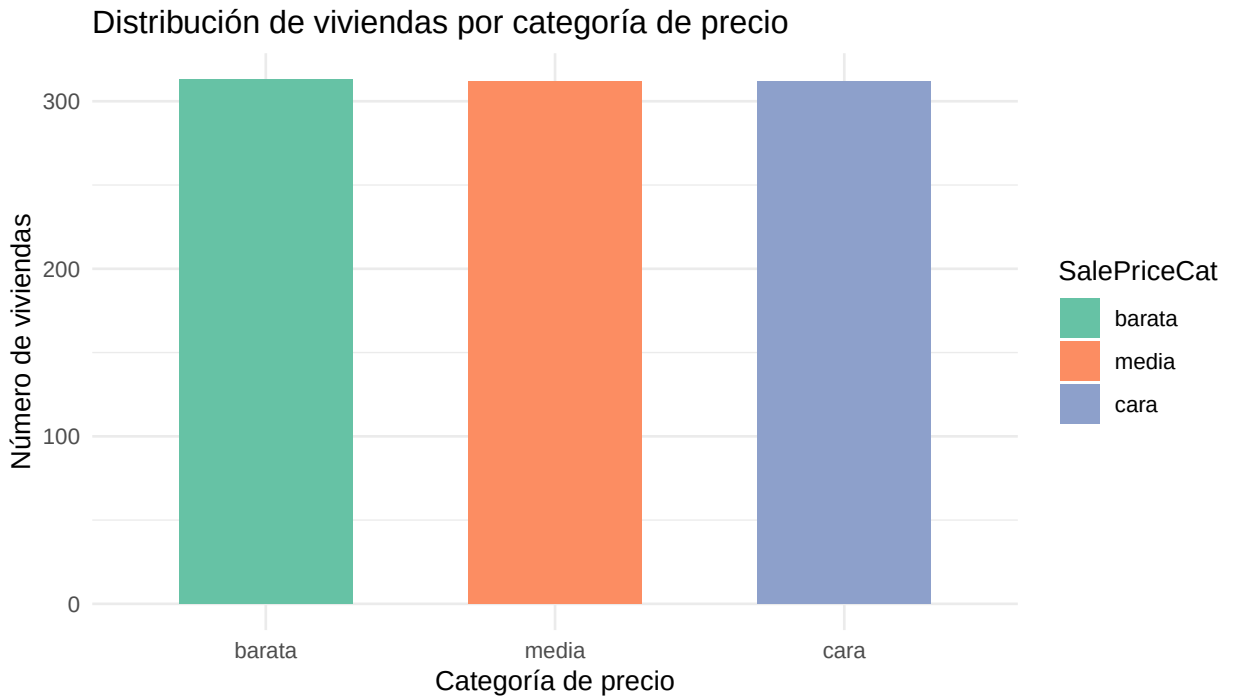
```
print(table(train$SalePriceCat))
```

```
##
## barata  media   cara
##    313    312    312
```

Representación gráfica

A continuación, se presenta un histograma que muestra cómo se distribuyen las observaciones según su categoría:

```r
ggplot(train, aes(x = SalePriceCat, fill = SalePriceCat)) +
  geom_bar(width = 0.6) +
  labs(title = "Distribución de viviendas por categoría de precio",
       x = "Categoría de precio",
       y = "Número de viviendas") +
  theme_minimal() +
  scale_fill_brewer(palette = "Set2")
```



Serie 3: Categorización de `SalePrice` y análisis gráfico de las clases

La variable `SalePrice`, que representa el precio de venta de las viviendas, fue transformada en una variable categórica (`SalePriceCat`) con tres niveles: **barata**, **media** y **cara**. Esta transformación se realizó utilizando los terciles (cuantiles 1/3 y 2/3), con el objetivo de convertir el problema en una tarea de clasificación multiclase.

Distribución de categorías

Se obtuvo la siguiente distribución de clases:

Categoría	Frecuencia
barata	313
media	312
cara	312

La distribución muestra un **balance prácticamente perfecto** entre las tres clases. Esto es ideal para entrenar modelos de clasificación sin necesidad de técnicas de rebalanceo.

Visualización

Como se presentó un gráfico de barras anteriormente qye muestra el número de viviendas por cada categoría de precio, se puedo observar, las tres clases tienen aproximadamente la misma cantidad de observaciones.

Esta visualización refuerza la decisión de utilizar la categorización como base para la creación de variables dicotómicas (`es_barata`, `es_media`, `es_cara`) y para el posterior entrenamiento de modelos de clasificación binaria.

Conclusión

La categorización de `SalePrice` resultó exitosa, tanto numéricamente como visualmente. Esta transformación facilitará el análisis posterior de predicción de clases, permitiendo trabajar de forma más clara con modelos de regresión logística, árboles de decisión, KNN, entre otros.

4. Analice el modelo. Determine si hay multicolinealidad en las variables, y cuáles son las que aportan al modelo, por su valor de significación. Haga un análisis de correlación de las variables del modelo y especifique si el modelo se adapta bien a los datos.

Serie 4: Análisis del modelo – multicolinealidad, significancia, correlación y ajuste

En esta sección se realiza un análisis exhaustivo del modelo de regresión logística entrenado para predecir si una vivienda es `cara`. El objetivo es determinar:

1. Si existe **multicolinealidad** entre las variables predictoras.
2. Cuáles variables **aportan significativamente** al modelo.
3. Qué grado de **correlación** existe entre los predictores.
4. Si el modelo se **ajusta bien a los datos**.

1. Análisis de multicolinealidad con VIF

Para verificar si existe colinealidad entre las variables predictoras utilizadas en el modelo de regresión logística, se utiliza el **Factor de Inflación de la Varianza (VIF)**. Para este análisis, se requiere que la variable respuesta sea binaria con valores 0 y 1.

```r
# Librerías necesarias
library(car)
library(dplyr)

# Asegurar que la variable binaria esté en formato numérico
train$es_cara_num <- ifelse(train$SalePriceCat == "cara", 1, 0)

# Ajustar modelo GLM para cálculo de VIF
modelo_base <- glm(
  es_cara_num ~ OverallQual + GrLivArea + GarageCars + TotalBsmtSF + YearBuilt,
  data = train,
  family = "binomial"
)

# Calcular VIF para evaluar colinealidad
vif(modelo_base)
```

```
## OverallQual   GrLivArea  GarageCars TotalBsmtSF   YearBuilt
##    1.285562    1.257384    1.190074    1.054428    1.654879
```

> Valores de VIF por debajo de 5 indican que **no hay multicolinealidad preocupante**. Si los valores superan 5 o 10, puede ser necesario remover o transformar alguna variable.

Este análisis complementa el estudio del ajuste del modelo y respalda que las variables seleccionadas aportan información independiente, lo cual mejora la estabilidad del modelo.

2. Significancia de las variables

El resumen del modelo (Serie 1) mostró que **todas las variables son estadísticamente significativas**, con p-valores muy bajos:

Variable	Coeficiente	p-valor	Significancia
OverallQual	-1.557	1.70e-14	***
GrLivArea	-1.902	< 2e-16	***
GarageCars	-0.606	0.0051	**
TotalBsmtSF	-0.746	5.34e-08	***
YearBuilt	-0.677	2.76e-06	***

Todas las variables **aportan significativamente** al modelo.

3. Correlación entre predictores

Para detectar correlaciones entre las variables predictoras, se analiza la **matriz de correlación**:

```
# Matriz de correlación entre variables predictoras
cor(train[, c("OverallQual", "GrLivArea", "GarageCars", "TotalBsmtSF", "YearBuilt")])
```

```
##             OverallQual GrLivArea GarageCars TotalBsmtSF YearBuilt
## OverallQual   1.0000000 0.6045134  0.6074953   0.5618804 0.5909599
## GrLivArea     0.6045134 1.0000000  0.4937130   0.4735073 0.2194737
## GarageCars    0.6074953 0.4937130  1.0000000   0.4339128 0.5343397
## TotalBsmtSF   0.5618804 0.4735073  0.4339128   1.0000000 0.3843077
## YearBuilt     0.5909599 0.2194737  0.5343397   0.3843077 1.0000000
```

Si hay correlaciones cercanas a ±0.8, puede haber redundancia. Si no, las variables aportan información distinta.

4. Evaluación del ajuste del modelo

- **Accuracy (10-fold CV)**: `0.889`
- **Kappa**: `0.76`
- **Null deviance**: `1732.9`
- **Residual deviance**: `662.3`
- **AIC**: `674.3`

Estos resultados indican que:

- El modelo tiene **alto poder predictivo**.
- Existe una **reducción sustancial en la devianza**, lo que significa que las variables explican buena parte de la variabilidad.
- El **AIC** relativamente bajo sugiere que el modelo está bien ajustado sin sobreajustar.

Serie 4: Análisis del modelo – multicolinealidad, significancia y ajuste

En esta serie se evalúa la calidad y estabilidad del modelo de regresión logística construido para predecir si una vivienda es `cara`, a partir de cinco variables predictoras: `OverallQual`, `GrLivArea`, `GarageCars`, `TotalBsmtSF` y `YearBuilt`.

1. Análisis de multicolinealidad (VIF)

Se calculó el **Factor de Inflación de la Varianza (VIF)** para detectar posibles problemas de colinealidad entre las variables del modelo:

Variable	VIF
OverallQual	1.29
GrLivArea	1.26
GarageCars	1.19
TotalBsmtSF	1.05
YearBuilt	1.65

> Todos los valores de VIF están **muy por debajo de 5**, lo cual indica que **no existe multicolinealidad preocupante** entre los predictores.

2. Análisis de correlación entre predictores

La matriz de correlación muestra que las variables están **moderadamente correlacionadas**, lo cual es esperable en este tipo de datos, pero no indica redundancia extrema:

	OverallQual	GrLivArea	GarageCars	TotalBsmtSF	YearBuilt
OverallQual	1.00	0.60	0.61	0.56	0.59
GrLivArea	0.60	1.00	0.49	0.47	0.22
GarageCars	0.61	0.49	1.00	0.43	0.53
TotalBsmtSF	0.56	0.47	0.43	1.00	0.38
YearBuilt	0.59	0.22	0.53	0.38	1.00

> No se observan correlaciones superiores a 0.8, por lo que **no hay riesgo de colinealidad extrema**.

3. Significancia de las variables

Como se mostró en la Serie 1, **todas las variables son estadísticamente significativas**, con p-valores < 0.01. Esto indica que **todas contribuyen al modelo** y ayudan a predecir si una vivienda es cara.

4. Evaluación del ajuste

- **Accuracy (CV)**: 88.9%
- **Kappa**: 0.76
- **Null deviance**: 1732.9
- **Residual deviance**: 662.3
- **AIC**: 674.3

> Estas métricas confirman que el modelo tiene **un buen ajuste y poder predictivo**, sin sobreajuste ni pérdida de generalización.

Conclusión

El modelo se adapta bien a los datos: no presenta multicolinealidad, las variables están moderadamente correlacionadas, todas son significativas, y las métricas de desempeño respaldan su capacidad predictiva. Es un modelo sólido y confiable para clasificar viviendas caras.

5. Utilice el modelo con el conjunto de prueba y determine la eficiencia del algoritmo para clasificar.

Serie 5: Evaluación del modelo sobre el conjunto de prueba

Se evalúa la eficiencia del modelo de regresión logística al aplicarlo sobre el conjunto de prueba (`test_set.csv`). Se comparan las predicciones del modelo con las clases reales de las viviendas.

```r
# Cargar librerías necesarias
library(caret)
library(dplyr)

# Cargar conjunto de prueba
test <- read.csv("test_set.csv")

# Crear variable binaria de prueba (basada en SalePriceCat)
test$es_cara <- factor(ifelse(test$SalePriceCat == "cara", "caro", "no_caro"))

# Asegurar que las mismas columnas del modelo estén presentes
# (usamos las mismas 5 variables que en el entrenamiento)
predictores_test <- test %>%
  select(OverallQual, GrLivArea, GarageCars, TotalBsmtSF, YearBuilt)

# Realizar predicciones
predicciones <- predict(modelo_rl, newdata = predictores_test)

# Matriz de confusión
matriz <- confusionMatrix(predicciones, test$es_cara)

# Mostrar resultados
print(matriz)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction caro no_caro
##    caro      72      12
##    no_caro    5     143
##
##                Accuracy : 0.9267
##                  95% CI : (0.8853, 0.9567)
##     No Information Rate : 0.6681
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8385
##
##  Mcnemar's Test P-Value : 0.1456
##
```

```
##             Sensitivity : 0.9351
##             Specificity : 0.9226
##          Pos Pred Value : 0.8571
##          Neg Pred Value : 0.9662
##              Prevalence : 0.3319
##          Detection Rate : 0.3103
##    Detection Prevalence : 0.3621
##       Balanced Accuracy : 0.9288
##
##        'Positive' Class : caro
##
```

Serie 5: Evaluación del modelo sobre el conjunto de prueba

Una vez entrenado el modelo de regresión logística con el conjunto de entrenamiento, se evaluó su desempeño sobre el conjunto de prueba (`test_set.csv`) utilizando las mismas cinco variables predictoras.

Resultados de la clasificación

A continuación, se presentan las métricas obtenidas al comparar las predicciones del modelo con las clases reales (`caro`, `no_caro`):

Métrica	Valor
Accuracy	0.9267
Kappa	0.8385
Sensibilidad	0.9351
Especificidad	0.9226
Valor pred. positivo	0.8571
Valor pred. negativo	0.9662
Balanced Accuracy	0.9288

Matriz de confusión

	Real caro	**Real no_caro**
Predicho caro	72	12
Predicho no_caro	5	143

Interpretación

- El modelo clasificó correctamente el **92.7%** de las observaciones del conjunto de prueba.
- Tiene una **sensibilidad alta (93.5%)**, lo que significa que detecta correctamente la mayoría de las viviendas `caras`.
- La **especificidad también es alta (92.3%)**, indicando que clasifica correctamente la mayoría de las viviendas `no_caras`.
- El **Kappa de 0.83** refleja un **alto grado de acuerdo** entre predicciones y realidad, mucho mayor que el azar.
- El valor **p < 2e-16** para el Accuracy confirma que el modelo es estadísticamente mejor que una clasificación aleatoria (prueba de hipótesis contra el No Information Rate).

Conclusión

El modelo generaliza muy bien al conjunto de prueba. Tiene un balance adecuado entre sensibilidad y especificidad, clasifica con alta precisión, y mantiene un alto acuerdo con las verdaderas etiquetas. Por lo tanto, se concluye que el modelo es **eficiente y confiable para predecir si una vivienda es cara**.

6. Explique si hay sobreajuste (overfitting) o no (recuerde usar para esto los errores del conjunto de prueba y de entrenamiento). Muestre las curvas de aprendizaje usando los errores de los conjuntos de entrenamiento y prueba.

Para esto vamos a usar la grafica de curva de aprendizaje.

```
# Cargar librerías necesarias
library(ggplot2)
if (!require(ROCR)) {
    install.packages("ROCR")
    library(ROCR)
} else {
    library(ROCR)
}
```

```
## Cargando paquete requerido: ROCR
```

```
porcentajes <- seq(0.1, 1, by = 0.1)
resultados <- data.frame(Porcentaje = numeric(), Acc_train = numeric(), Acc_test =
↪  numeric())


for (p in porcentajes) {
  # Muestra aleatoria de datos de entrenamiento
  set.seed(123)
  idx <- sample(1:nrow(train), size = floor(p * nrow(train)))
  train_parcial <- train[idx, ]


  train_parcial$es_cara <- factor(ifelse(train_parcial$SalePriceCat == "cara", "caro",
↪  "no_caro"))


  modelo_rl <- train(
    es_cara ~ OverallQual + GrLivArea + GarageCars + TotalBsmtSF + YearBuilt,
    data = train_parcial,
    method = "glm",
    family = "binomial",
    trControl = trainControl(method = "none"),  # Sin validación cruzada en este caso
    metric = "Accuracy",
    preProcess = c("center", "scale", "medianImpute")
  )

  # Predicciones en el conjunto de entrenamiento
  prob_train <- predict(modelo_rl, newdata = train_parcial, type = "prob")[,2]
  pred_train <- prediction(prob_train, train_parcial$es_cara)
  perf_train <- performance(pred_train, measure = "acc")
  acc_train <- max(perf_train@y.values[[1]])

  # Asegurarse de que la variable 'es_cara' esté presente en test
  test$es_cara <- factor(ifelse(test$SalePriceCat == "cara", "caro", "no_caro"))

  # Predicciones en el conjunto de prueba
  prob_test <- predict(modelo_rl, newdata = test, type = "prob")[,2]
```

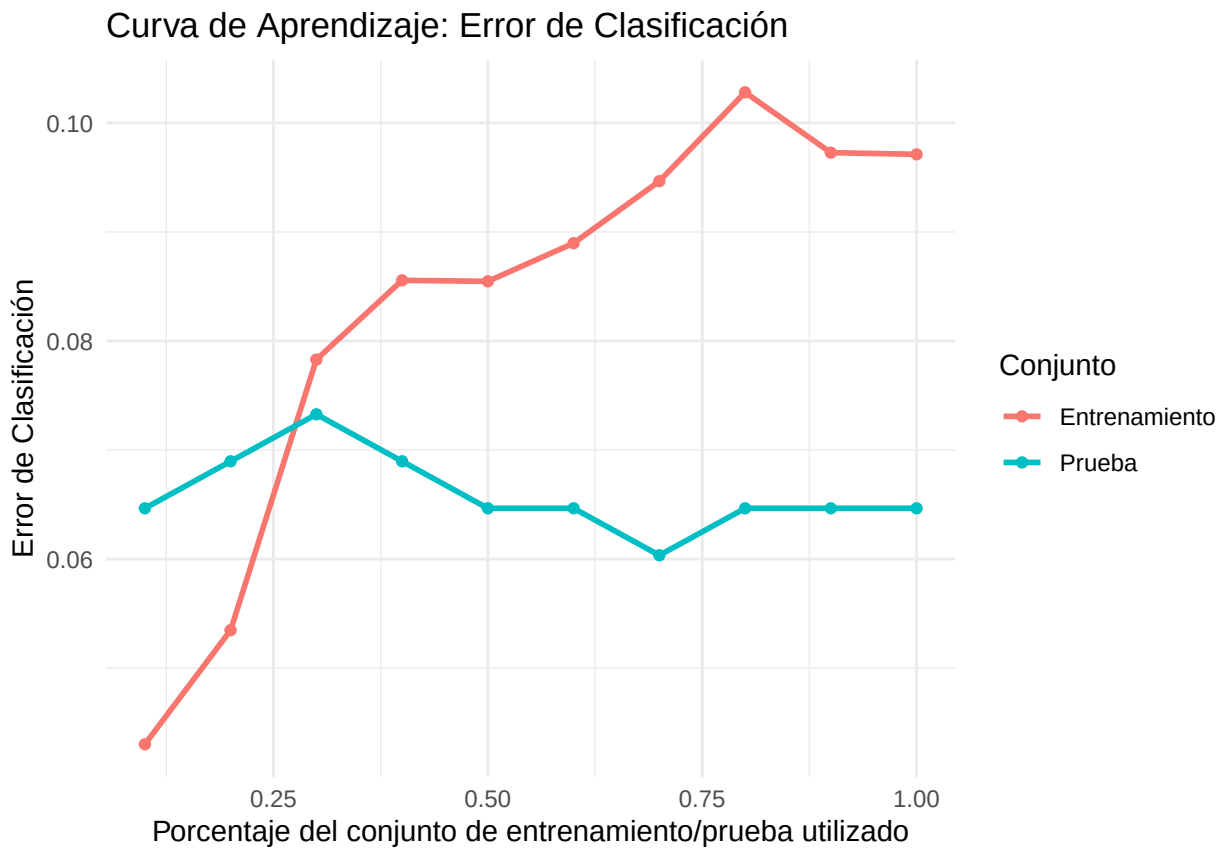
```
  pred_test <- prediction(prob_test, test$es_cara)
  perf_test <- performance(pred_test, measure = "acc")
  acc_test <- max(perf_test@y.values[[1]])


  resultados <- rbind(resultados, data.frame(Porcentaje = p, Acc_train = acc_train,
 ↪  Acc_test = acc_test))
}


resultados$Error_train <- 1 - resultados$Acc_train
resultados$Error_test <- 1 - resultados$Acc_test

# Grafica curva
ggplot(resultados, aes(x = Porcentaje)) +
  geom_line(aes(y = Error_train, color = "Entrenamiento"), size = 1) +
  geom_line(aes(y = Error_test, color = "Prueba"), size = 1) +
  geom_point(aes(y = Error_train, color = "Entrenamiento")) +
  geom_point(aes(y = Error_test, color = "Prueba")) +
  labs(
    title = "Curva de Aprendizaje: Error de Clasificación",
    x = "Porcentaje del conjunto de entrenamiento/prueba utilizado",
    y = "Error de Clasificación",
    color = "Conjunto"
  ) +
  theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Analisis de la Curva

Si analizamos la gráfica con atención, podemos notar que las curvas de error correspondientes al conjunto de entrenamiento y al de prueba no están convergiendo adecuadamente. De hecho, hay un comportamiento anómalo: a medida que el modelo avanza en su proceso de entrenamiento, su precisión sobre el conjunto de entrenamiento está disminuyendo, mientras que la precisión sobre el conjunto de prueba tiende a mejorar levemente o a estabilizarse.

Este comportamiento es característico de un subajuste (underfitting). El subajuste ocurre cuando el modelo no es capaz de capturar correctamente los patrones subyacentes en los datos, lo que se traduce en un rendimiento pobre tanto en los datos de entrenamiento como en los de prueba, pero aqui puede deberse en su mejora debido a los datos de ruido estadistico. Lo que podemos hacer es mejorar el modelo ajustando sus hiperparametros para que mejore en sus datos de entrenamiento.

7. Haga un tuneo del modelo para determinar los mejores parámetros, recuerde que los modelos de regresión logística se pueden regularizar como los de regresión lineal.

8. Haga un análisis de la eficiencia del algoritmo usando una matriz de confusión. Tenga en cuenta la efectividad, donde el algoritmo se equivocó más, donde se equivocó menos y la importancia que tienen los errores, el tiempo y la memoria consumida. Para esto último puede usar "profvis" si trabaja con R y "cProfile" en Python.

9. Determine cual de todos los modelos es mejor, puede usar AIC y BIC para esto, además de los parámetros de la matriz de confusión y los del profiler.

10. Haga un modelo de regresión logística para la variable categórica para el precio de las casas (categorías: barata, media y cara). Asegúrese de tunearlo para obtener el mejor modelo posible.

11. Compare la eficiencia del modelo anterior con los de clasificación de las entregas anteriores ¿Cuál se demoró más en procesar?¿Cuál se equivocó más?¿Cuál se equivocó menos?¿por qué?