Minería de Datos y Modelización Predictiva (I)

Fernández Hernández Alberto

12/28/2020

1. Depuración de los datos

Inicialmente nos encontramos ante un conjunto de datos con la información demográfica de los diferentes municipios en España así como sus últimos resultados electorales. Por ello, antes de elegir las variables objetivo y elaborar el modelo de regresión, echemos un primer vistazo a los datos proporcionados. En primer lugar, y antes de analizar las variables independientes, debemos **recategorizar las variables objetivo cualitativas como factor**, dado que el formato establecido por defecto es numérico. Sin contar el campo *CodigoProvincia* (el cual mencionaremos más adelante) existen un total de 6 variables categóricas, incluyendo las variables objetivo cualitativas (AbstencionAlta, Izquierda y Derecha), además de los campos CCAA, ActividadPpal y Densidad, dado que contienen un número limitado de valores únicos:

```
# c(3, 7, 11, 12, 34, 38) => CCAA, AbstencionAlta, Izquierda, Derecha, ActividadPpal y Densidad
datos[,c(3, 7, 11, 12, 34, 38)] <- lapply(datos[,c(3, 7, 11, 12, 34, 38)], factor)
```

Por otro lado, podemos observar que los datos proporcionados contienen un total de 41 variables, de las cuales cabe destacar el campo identificador *Name*, un campo con el nombre de cada municipio:

```
## Nombres de municipio unicos:  8102  de  8119 filas
```

Salvo excepciones, en las que el nombre del municipio coincide, se trata de un campo que podríamos considerar como identificativo, por lo que no nos aportará información relevante al modelo y por ello lo eliminamos:

```
datos <- datos[, -c(1)] # Eliminamos ademas las variables objetivo que no vayamos a emplear
```

Por otro lado, nos encontramos con el campo *CodigoProvincia* que, a diferencia del anterior, el número de valores diferentes es significativamente menor (52 valores únicos). No obstante, nos encontramos ante la siguiente duda ¿Mantenemos los datos en formato numérico o los recategorizamos a *factor*? Por un lado, recategorizarlo como una variable cualitativa puede llegar a entorpecer la elaboración del modelo, en especial si una o varias de las categorías no está lo suficientemente representada y debe ser agrupada. Por otro lado, si queremos mantener la variable como un campo numérico tiene que aportar "sentido" al modelo, esto es, asegurar que la media del código de provincia es de 26.67, por ejemplo, no aporta información, además de que no tiene sentido hablar de media o mediana en este campo. Por otro lado, si quisiéramos emplearlo como un campo más, deberá aportar un cierto grado de importancia, tanto a las variables cuantitativas como cualitativas:

```
# Columnas 5,7,8 y 9 correspondientes con las variables objetivo cuantitativas
cor(cbind(datos$CodigoProvincia, datos[, c(5,7,8,9)]), use = "complete.obs", method = "pearson")[1,-1]
```

```
## AbstentionPtge       Izda_Pct       Dcha_Pct      Otros_Pct
##    -0.05711613    -0.01653135     0.12657294    -0.09079252
```

```
# Mediante la V Cramer vemos la importancia sobre las variables objetivo cualitativas
salida <- c()
for(col in c("AbstencionAlta", "Izquierda", "Derecha")) {
  salida <- cbind(salida, sapply(datos[ , "CodigoProvincia"],function(x) Vcramer(x,unlist(datos[, col]))))
}
salida
```

```
##                       [,1]      [,2]      [,3]
## CodigoProvincia 0.1508031 0.1609849 0.2371896
```

Como podemos observar en ambas salidas, la correlación tanto en las variables cuantitativas como cualitativas no superan el 0.12 y 0.23 respectivamente, valores bastante bajos, por lo que podemos descartar la variables para ambos modelos. Antes de continuar, de cara a valorar la calidad de la depuración final guardamos en una variable los valores de correlación originales, con el objetivo de compararlos con los del conjunto de datos ya depurado:

```
corr.previa <- cor(datos[,unlist(lapply(datos, is.numeric))], use="complete.obs", method="pearson")
```

1.1 Valores erróneos / no declarados

A continuación, procedemos a eliminar aquellos valores no declarados en las variables, así como posible valores fuera de rango:

1. *ForeignersPtge* negativos. Porcentajes de extranjeros menores a cero:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
##   -8.96    1.06    3.59    5.62    8.18   71.47
```

```
datos$ForeignersPtge<-replace(datos$ForeignersPtge, which(datos$ForeignersPtge < 0), NA)  # Min < 0
```

2. Porcentajes de *SameComAutonPtge* y *PobChange_pct* superiores al 100 % (en el caso de *PobChange_pct* según la documentación es posible la aparición de porcentajes negativos, pero no superiores al 100 %):

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   75.81   84.49   81.63   90.46  127.16
```

```
datos$SameComAutonPtge <-replace(datos$SameComAutonPtge, which(datos$SameComAutonPtge > 100), NA)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## -52.2700 -10.4000  -4.9600  -4.8974   0.0925 138.4600        7
```

```
datos$PobChange_pct <-replace(datos$PobChange_pct, which(datos$PobChange_pct > 100), NA) # Max > 100
```

3. Valores a 99999 en la columna *Explotaciones*, posible indicativo de la ausencia de valores en estos casos:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1      22      52    2447     137   99999
```

```
datos$Explotaciones<-replace(datos$Explotaciones,which(datos$Explotaciones==99999),NA) # Max == 99999
```

4. Categoría "?" sin declarar en *Densidad*, por lo que lo recategorizamos a *NA*:

```
##         ?  Alta Baja MuyBaja
## n    92.0 557.0 1053    6417
## %     1.1   6.9   13      79
## val%  1.1   6.9   13      79
```

```
datos$Densidad<-recode.na(datos$Densidad,"?")
```

```
## Recoded 92 values to NA.
```

1.2 Análisis de valores atípicos

Una vez corregidos los errores detectados, analicemos los valores atípicos más destacados empleando la función *describe*:

```
##                          sd  skew kurtosis
## Population          46215.20 45.98  2814.43
## TotalCensus         34428.89 46.49  2888.34
## totalEmpresas        4219.37 53.68  3472.00
## ComercTTEHosteleria  1233.02 45.40  2646.94
## Servicios            2446.81 57.48  3830.74
## inmuebles           24314.71 44.53  2643.65
## Pob2010             47535.68 47.15  2939.56
## SUPERFICIE           9218.19  6.07    62.28
```

Como podemos observar en la salida anterior, las columnas con la población, el censo total, el número total de empresas, así como la superficie son los que mayor desviación presentan con respecto a su media, lo que se traduce además de un coeficiente de simetría mayor a 1 (asimetría) y una elevada curtosis, claros indicios de la presencia de valores atípicos (algo lógico si nos planteamos el caso de la Población, con municipios que no superan el millar de habitantes en contraste con grandes municipios como Madrid o Barcelona). Por ello, comenzamos analizando el porcentaje maximo de valores atipicos en nuestro conjunto de datos:

```
## Servicios
##  11.87338
```

En este caso, el maximo porcentaje corresponde con el campo *Servicios*, con un 11.87 %, una de las variables con elevada desviación típica que habíamos planteado previamente. Por ello, dado que el máximo porcentaje de valores atípicos no supera el 12 % podemos marcalos como ausentes sin problema alguno para posteriormente imputarlos.

1.3 Análisis de valores missings (NA). Imputaciones

Tras recodificar los valores atípicos como ausentes, debemos analizar la proporción de valores atípicos tanto por observacion como por variable. Para ello, obtenemos el valor maximo de atípicos tanto por fila como por columna:

```
##   Por.observacion Por.variable
## 1            37.5     12.63702
```

Aparentemente, mientras que el porcentaje de *missings* por variable es del 12.64 %, por observaciones detectamos un mayor numero (37.5 %). No obstante, si empleamos la función *summary*:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   0.000   0.000   3.874   3.125  37.500
```

Vemos que el 75 % de las observaciones contienen aproximadamente un 3 % de valores *missings* o menos, por lo que no parece tratarse de un elevado número de filas. Por otro lado, la pérdida de información en ambos casos no supera el 50 %, por lo que en lugar de eliminar dichos valores podemos imputarlos. De forma previa a la imputación, existen determinados campos que pueden ser imputados sin necesidad de emplear una media, mediana o de forma aleatoria:

1. *Age_19_65_pct*, cuyo porcentaje de edad puede calcularse a partir de la suma de *Age_under19_Ptge* y *Age_over65_pct*:

```
x["Age_19_65_pct"] <- 100 - (as.numeric(x["Age_under19_Ptge"]) + as.numeric(x["Age_over65_pct"]))
```

2. *totalEmpresas*, cuyo porcentaje puede obtenerse a partir de la suma de *Industria*, *Construccion*, *ComercTTEHosteleria* y *Servicios*:

```
x["totalEmpresas"] <- as.numeric(x["Industria"]) + as.numeric(x["Construccion"]) +
                      as.numeric(x["ComercTTEHosteleria"]) + as.numeric(x["Servicios"])
```

3. *Densidad*, cuyo valor puede obtenerse a través del cociente entre *Population* y *SUPERFICIE*: si la proporcion es menor o igual a 1 decimos que la densidad es "MuyBaja"; si está entre 1 y 5 decimos que es "Baja" y si es mayor o igual a 5 diremos que es "Alta"

```
ifelse(proporcion < 1, densidad <- "MuyBaja", ifelse(proporcion > 1 & proporcion < 5,
      densidad <- "Baja", densidad <- "Alta"))
```

Para el proceso de imputación en el resto de variables se ha dividido en un total de dos fases por el siguiente motivo: hay demasiados valores *missing* consecutivos. Esto supone que, a la hora de realizar la imputación con valores aleatorios (mediante una interpolación) muchos de los valores *missing* quedan sin imputarse, dado que muchos de ellos parecen ser consecutivos, lo que impide calcular su valor. Para ello, se realiza una primera imputación de forma aleatoria para, a continuación, imputar los valores restantes mediante la mediana (dado que muchos de los valores atípicos marcados a *missing* presentaban una elevada desviación típica, lo que hace que la mediana sea mucho más representativa que la media):

```
## 10064  valores atipicos
```

```
datos[,columnas] <- sapply(datos[, columnas],function(x) ImputacionCuant(x,"aleatorio"))
```

```
## 1219  valores atipicos tras la imputacion aleatoria
```

```
datos[,columnas] <- sapply(datos[, columnas],function(x) ImputacionCuant(x,"mediana"))
```

```
## 974  valores atipicos tras la imputacion con la mediana
```

Como podemos observar, hemos conseguido reducir el porcentaje de *missings*. A continuación, si imputamos las tres columnas mencionadas anteriormente a partir del resto de variables:

```
## 0  valores atipicos
```

Tras la imputación final, veamos qué porcentaje de correlación se ha perdido comparando la correlación del conjunto de datos inicial con respecto al conjunto de datos depurado:

```r
corr.posterior <- cor(datos[,unlist(lapply(datos, is.numeric))], use = "complete.obs" , method="pearson")
comparacion.corr <- corr.posterior[-33, -33] - corr.previa # Eliminamos el campo prop_missings
sum(abs(comparacion.corr) < 0.2) * 100 / (dim(comparacion.corr)[1] * dim(comparacion.corr)[2])
```

```
## [1] 82.36915
```

Podemos observar que aproximadamente un 82.55 % de las variables se ha visto reducida en menos de 0.2 con respecto a la correlación original, bastante mayor con respecto a una imputación únicamente con la media o con la mediana. Tras eliminar los valores *missing* debemos preguntarnos ¿Qué variables escojo como variables objetivo? Para ello, se ha empleado como criterio **aquella variable objetivo** con mayor suma de correlaciones con respecto al resto de variables:

```
## AbstentionPtge . Suma correlaciones:  3.323285
## Izda_Pct . Suma correlaciones:  4.732928
## Dcha_Pct . Suma correlaciones:  7.698831
## Otros_Pct . Suma correlaciones:  8.567008
```

```
## AbstencionAlta . Suma correlaciones:  4.926981
## Izquierda . Suma correlaciones:  5.435022
## Derecha . Suma correlaciones:  8.084765
```

Comenzando con las variables cuantitativas, en un principio elegiríamos como variable objetivo *Otros_Pct*, de no ser por un inconveniente: el elevado número de valores atípicos que presenta, lo que puede llegar a dificultar la elaboración del modelo:

```r
sapply(Filter(is.numeric, datos[, c(4:8)]),function(x) atipicosAmissing(x)[[2]]) * 100/nrow(datos)
```

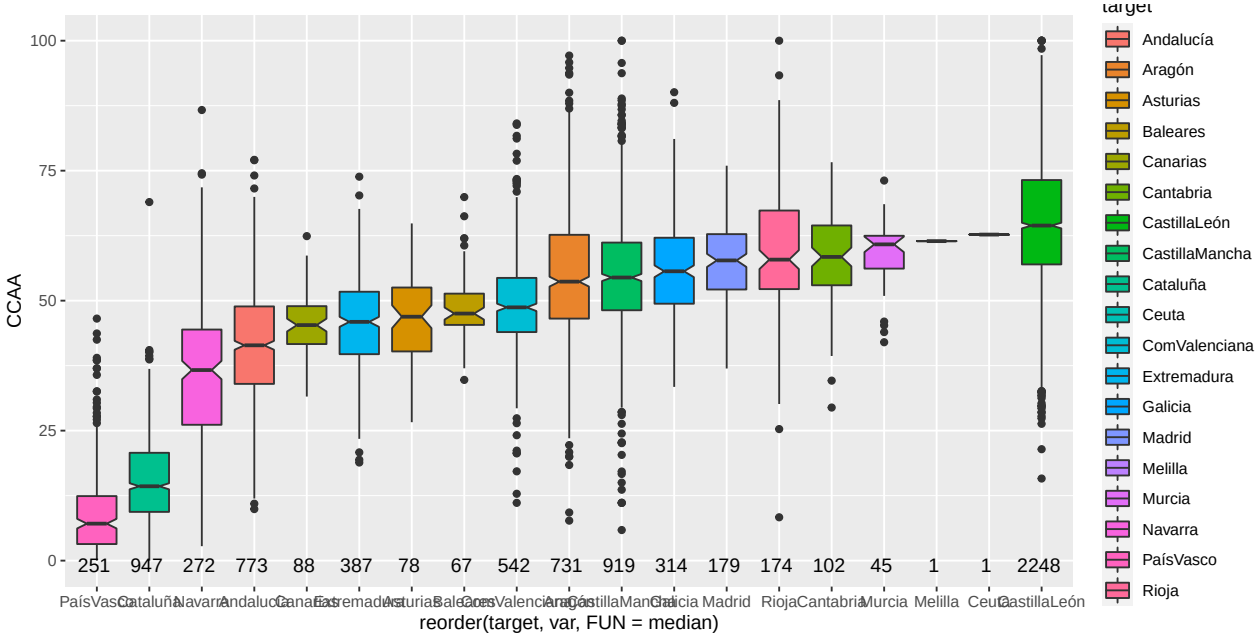
```
## AbstentionPtge       Izda_Pct       Dcha_Pct      Otros_Pct
##        0.00000        0.00000        0.00000       10.40769
```

Por ello, y con el objetivo de facilitar el desarrollo de un modelo lineal elegimos el porcentaje de votos a la derecha o *Dcha_Pct*, el cual también presenta un porcentaje de correlación elevado con respecto al resto de variables. En relación a la variable binaria, dada la elevada correlación acumulada que presenta, elegimos como variable objetivo binaria el campo *Derecha*. El resto de variables objetivo son eliminadas.

Tras eliminar el resto de variables, debemos hacernos la siguiente pregunta. De las variables cualitativas ¿Podemos agrupar alguna de sus categorías? Salvo el campo *Densidad*, donde la frecuencia de cada categoría está repartida de forma equitativa:

```
##      MuyBaja Baja  Alta
## n     6509.0 1053 557.0
## %       80.2   13   6.9
## val%    80.2   13   6.9
```

Tanto en el campo *CCAA* como *ActividadPpal* debemos agrupar algunas de las categorías. Comenzando con las Comunidades Autónomas, dado que disponemos de 19 valores diferentes (algunos de los cuales como Ceuta o Melilla con una única representación), tal y como se muestra a continuación:



Para agrupar las Comunidades Autónomas, no solo agruparemos aquellas categorías con un menor número de variables sino además aquellas Comunidades **cuya amplitud en el diagrama de caja y bigotes sea similar**: **País Vasco y Cataluña**, **Navarra y Andalucía**, **ComValenciana, Extremadura, Asturias, Baleares y Canarias**, **Aragón y Castilla la Mancha**, así como **Galicia, Cantabria, Madrid, La Rioja, Ceuta, Melilla y Murcia'**. En el caso de Castilla y León, dado que se trata de la CCAA con mayor número de observaciones, no la agruparemos con otra comunidad. No obstante, de cara a la creación de los modelos es importante tener en cuenta que se trata de la CCAA con la mayor distribución de votos hacia la derecha, además de ser la única categoría que no ha sido agrupada, por lo que lo consideraremos como la **categoría de referencia**, recodificando su nombre a *AA_CL* (de esta manera la categoría será elegida como referencia por orden alfabético).

En contraposición, nos encontramos con el campo *ActividadPpal*:

```
##      ComercTTEHosteleria Construccion Industria   Otro Servicios
## n                 2540.0         14.0      13.0 4932.0     620.0
## %                   31.3          0.2       0.2   60.7       7.6
## val%                31.3          0.2       0.2   60.7       7.6
```

En este campo, nos encontramos con los campo *Construccion* e *Industria* con apenas 14 y 13 apariciones, respectivamente. Por ello, dado que solo hay que agrupar dos categorías con poca representación (a diferencia de las CCAA donde teníamos 19), lo agruparemos con *Servicios*, la siguiente categoría con menor número de apariciones:

```
datos$ActividadPpal <- recode(datos$ActividadPpal,
                      "c('Construccion', 'Industria', 'Otro') = 'Construccion_Industria_Otro';")
```

1.4 Transformaciones de variables y relaciones con las variables objetivo

Una vez recategoriazadas las variables, puede ser necesario realizar alguna transformación en las variables para poder plasmar de este modo la verdadera relación de las variables independientes con la variable objetivo:

```
input_cont<-data.frame(varObjCont,datos,Transf_Auto(Filter(is.numeric, datos),varObjCont))
input_bin<-data.frame(varObjBin,datos,Transf_Auto(Filter(is.numeric, datos),varObjBin))
```

Sin embargo, debemos recordar un detalle fundamental: **cuantas más variables empleemos para elaborar nuestros modelos, más costoso (computacionalmente) será obtenerlo**. Por tanto, de todas las transformaciones que hemos obtenido ¿Algunas de ellas mejoran la correlación con las variables objetivo? Comenzando con la variable objetivo continua, **compararemos las dos matrices de correlación: las**

variables originales y sus transformadas. Sobre la diferencia filtraremos aquellas cuya diferencia sea igual o inferior a 0.01 (prácticamente idénticas):

```
correlaciones  <- round(cor(Filter(is.numeric, input_cont), use="pairwise", method="pearson")[1,32:61]
                        - cor(Filter(is.numeric, input_cont), use="pairwise", method="pearson")[1,2:31], 2)
# Filtramos aquellas variables transformadas que apenas hayan aumentado su correlacion
colnames(input_cont)[colnames(input_cont) %in% names(correlaciones[correlaciones > -0.01])]
```

```
##  [1] "xAge_under19_Ptge"         "sqrtxAge_19_65_pct"
##  [3] "xAge_over65_pct"           "xWomanPopulationPtge"
##  [5] "expxSameComAutonPtge"      "xSameComAutonDiffProvPtge"
##  [7] "logxSUPERFICIE"            "xPersonasInmueble"
##  [9] "sqrtxExplotaciones"        "sqrtxprop_missings"
```

Como podemos observar, 10 de las transformadas no aportan apenas mejoría al modelo. Por tanto eliminamos dichas transformadas, junto con aquellas variables originales que aportan menor correlación con respecto a sus transformadas.

Con respecto a la variable objetivo binaria, emplearemos el criterio del **Valor de la información** o *Information Value*, criterio que nos permite medir la capacidad predictiva que presenta una variable (agrupada en intervalos) para predecir con precisión los valores 1,0 de la variable objetivo. El objetivo será calcular la diferencia entre el *Information Value* de cada una de las variables originales (agrupadas por intervalos) con respecto a las variables transformadas:

```
salida.woe <- woebin(input_bin, "varObjBin", print_step = 0)
```

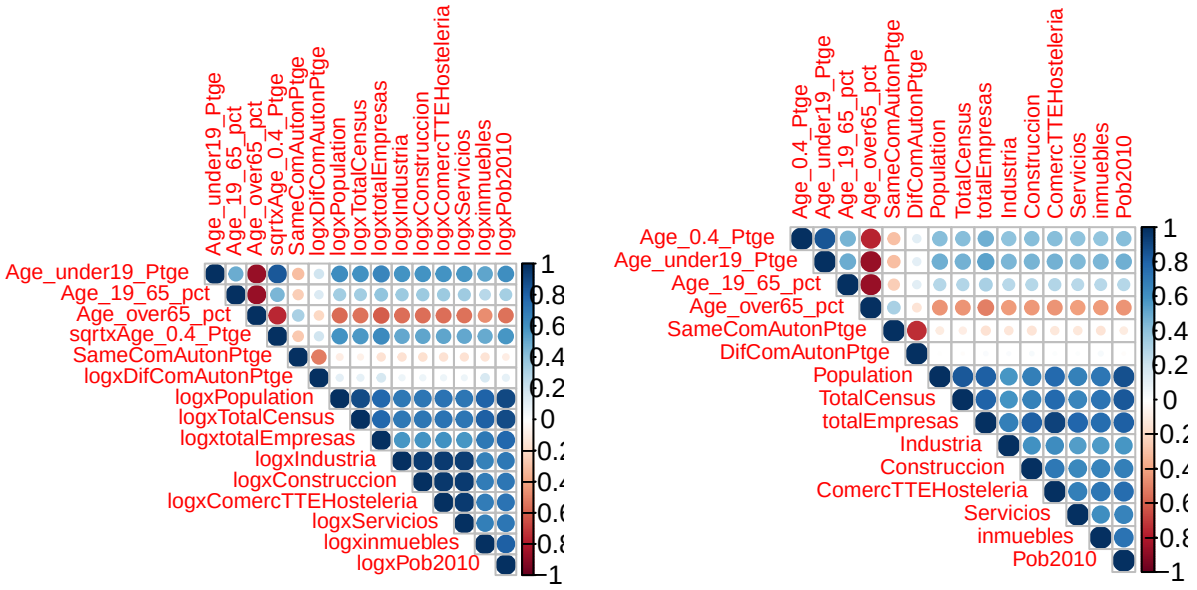
```
summary(sapply(salida.woe[c(2:24,26:28,30:33)], function(x) x$total_iv[1]) -
          sapply(salida.woe[c(34:63)], function(x) x$total_iv[1]))
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -1.787e-02 -1.927e-03 -9.045e-06  5.670e-04  2.433e-03  1.485e-02
```

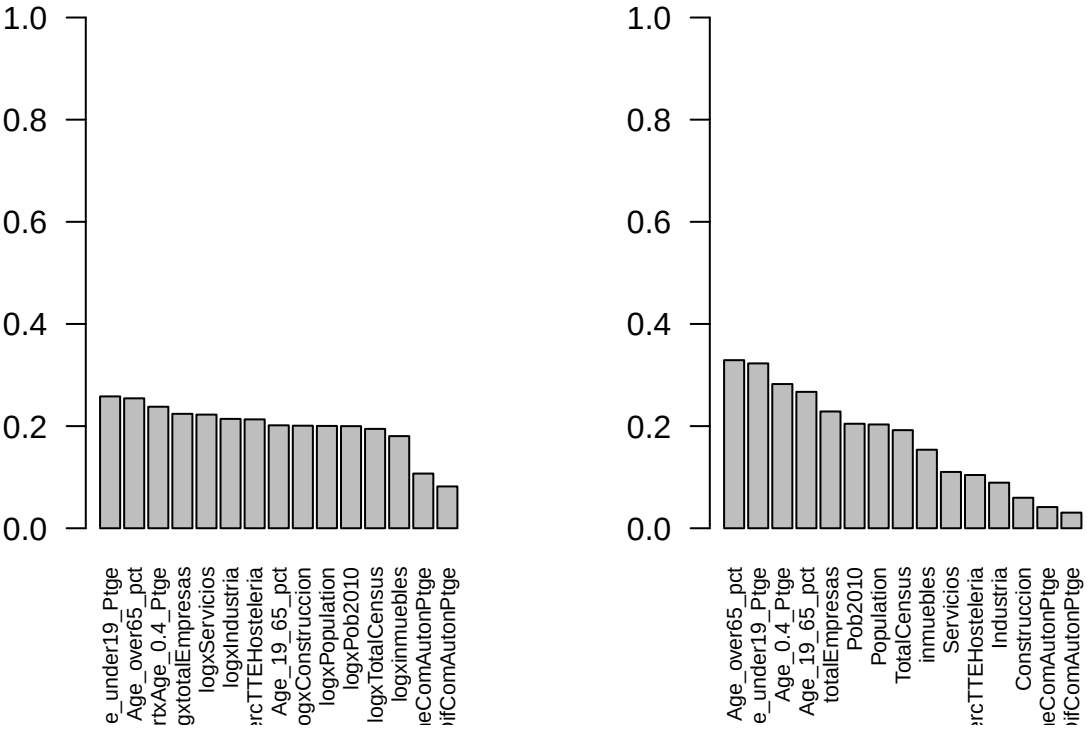
Si nos fijamos en la salida anterior, prácticamente ninguna de las variables transformadas mejora significativamente con respecto a la variable original, donde en el mejor de los casos el aporte máximo al valor de información es de 0.015. Por otro lado, el tercer cuartil nos indica que el 75 % de las variables ve mejorado su valor de información en 0.0025 o menos, por lo que apenas supone una mejoría. Por tanto, para el modelo de regresión logístico **mantendremos las variables origiales** (eliminamos las transformadas).

1.5 Detección de las relaciones entre las variables input y objetivo

Como último paso en el proceso de depuración, debemos analizar qué relación existen entre las variables independientes y objetivo, pero incluso algo mucho más importante ¿Hay correlación entre las variables independientes? Esto último debe tenerse en cuenta, dado que una alta colinealidad puede reducir significativamente la calidad de ambos modelos. Para ello, quisiera destacar cuatro casos en los que se produce colinealidad:



Tanto en las variables originales como incluso con aquellas transformadas, nos encontramos con grupos de variables que presentan una elevada correlación entre sí: **los grupos de edad** (entre 0 y 4 años, menor a 19, entre 19 y 65 y más de 65 años); **si residen o no en la misma CCAA** (*SameComAutomPtge* y *DifComAutonPtge*); el total de empresas con respecto al número de empresas de cada sector (Industria, Construcción, Comercio, Servicios, así como el número de inmuebles y la población en el año 2010); e incluso una elevada correlación entre el campo *Population* y *totalCensus*. Para solventar el problema, empleando la **V de Cramer** escojemos aquellas variables de cada grupo que tenga una mayor relevancia con respecto a la variable objetivo, mientras que el resto las eliminamos:



Como podemos observar en los gráfico anteriores, para el modelo lineal *input_cont* nos quedaremos con las variables *Age_under_19_Ptge* y *Age_19_65_pct* (no están muy correlacionadas entre sí), *SameComAutonPtge*, *logxPopulation* y *logxtotalEmpresas*. Con respecto a *input_bin*, conservaremos *Age_over65_pct*,

SameComAutonPtge, *Population* y *totalEmpresas*.

2. Construcción del modelo de regresion lineal

Una vez realizada la depuración de los datos, procedemos a elaborar el modelo de regresión lineal. En primera instancia, **ejecutaremos un primer modelo con todas las variables y todas sus transformaciones**, con el objetivo de echar un primer vistazo al modelo inicial (aunque no sea el definitivo):

```
formInt<-formulaInteracciones(input_cont,1)
modelo1<-lm(formInt,data=data_train)
# Funcion que devuelve los valores Train,Test,AIC y SBC
mostrar.estadisticas(modelo1, data_train, data_test, "lm", "varObjCont")
```

```
## Train:  0.7497595 ; Test:  0.7232902 ; Dif. (Train-Test):  0.02646931 ; AIC: 48773.99 ; SBC:  50367.04
## Numero de variables:  234
```

Inicialmente, con 234 variables nos encontramos con un modelo con valores de AIC y BIC bastante elevados, aunque con una diferencia entre el train y test de solo 0.02. Esto supone, en primer lugar, que no todas variables presentan la misma importancia en el modelo (incluso puede que muchas de ellas no aporten prácticamente información). Para comprobarlo, analizaremos las estadísticas de cada variable empleando la función *modelEffectSizes*:

```
variacion.r2 <- modelEffectSizes(modelo1, Print = FALSE)
summary(variacion.r2$Effects[, 4])
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
## 0.0000001 0.0000323 0.0001193 0.0002990 0.0003239 0.0074796         1
```

Analizando la salida anterior, podemos comprobar que el 75 % de las variables del modelo inicial presentan una importancia en el R2 de 0.0003 o menos, es decir, existe solo un 25 % de las variables que aportan más de 0.0003, por lo que debemos centrarnos en estas últimas (de cara a facilitar la selección clásica). Por ello, elegimos del modelo original aquellas variables atípicas (aportan más de 0.0003 al R2):

```
variables.mas.imp <- names(boxplot(variacion.r2$Effects[, 4], plot = FALSE)$out)
variables.mas.imp
```

```
## [1] "CCAA"                    "CCAA:Age_under19_Ptge"
## [3] "CCAA:SameComAutonPtge"   "CCAA:ActividadPpal"
## [5] "CCAA:PersonasInmueble"   "CCAA:logxForeignersPtge"
```

Muchas de las variables con mayor importancia corresponden con interacciones con el campo CCAA. Por tanto, de cara a un segundo modelo conservaremos las transformaciones más importantes junto con las columnas originales, dado que algunas de las variables pueden proporcionar más información al modelo si no están incluidas en ninguna interacción:

```
formInt <- paste0("varObjCont~",paste0(colnames(input_cont[-1]), collapse = "+"),"+",
                  paste0(variables.mas.imp, collapse = "+"))
modelo1.2<-lm(as.formula(formInt),data=data_train)
mostrar.estadisticas(modelo1.2, data_train, data_test, "lm", "varObjCont")
```

```
## Train:  0.7304779 ; Test:  0.725188 ; Dif. (Train-Test):  0.00528988 ; AIC: 48908.17 ; SBC:  49321.69
## Numero de variables:  60
```

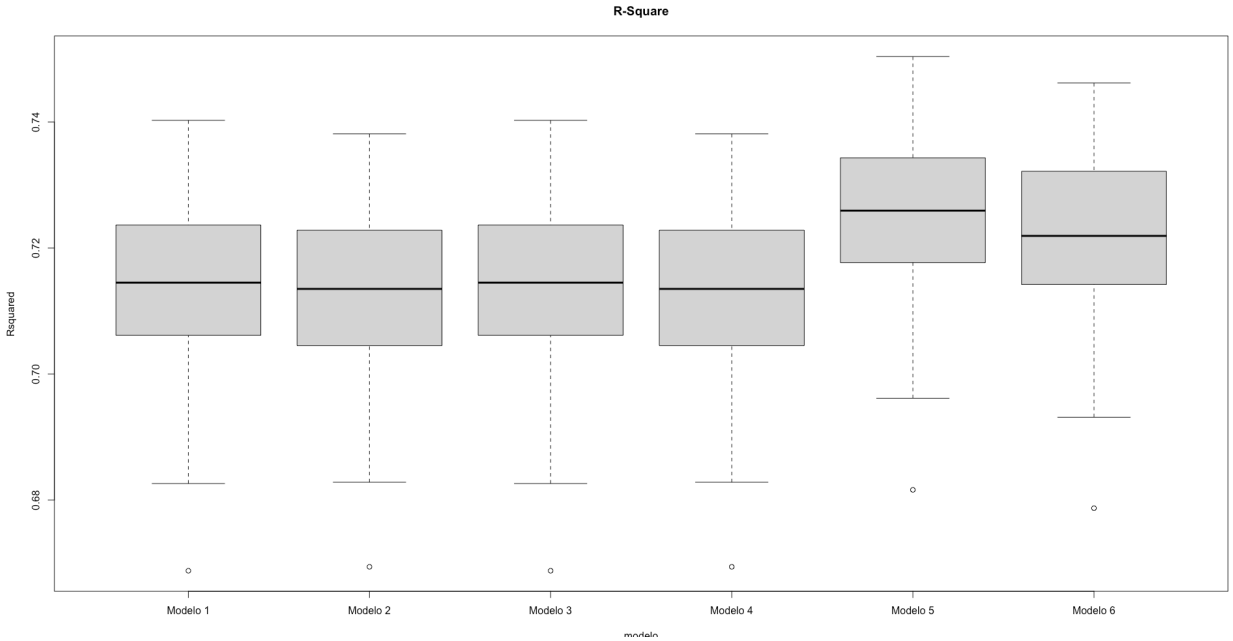
Reduciendo el número de parámetros a 60, pese a aumentar el AIC es más de 100 puntos, el criterio SBC consigue verse reducido en más de 1000 puntos, además de recortar la diferencia entre el R2 obtenido en los datos de entrenamiento y *test*. Con la formula del segundo modelo, podemos partir como base para la selección de variables clásica:

```
estadisticas.modelos <- seleccion.clasica(formInt, data_train, data_test, "lm")
```

```
##              R.2.train  R.2.test  Diferencia      AIC      SBC N.Parametros
## AIC-both     0.7180948 0.7144312 0.003663604 49155.98 49420.36           38
## SBC-both     0.7163374 0.7145427 0.001794747 49180.35 49390.50           30
```

```
## AIC-forward  0.7180948 0.7144312 0.003663604 49155.98 49420.36           38
## SBC-forward  0.7163374 0.7145427 0.001794747 49180.35 49390.50           30
## AIC-backward 0.7298823 0.7253877 0.004494548 48890.52 49195.57           44
## SBC-backward 0.7249686 0.7226587 0.002309877 48965.62 49128.32           23
```

Una vez calculada la selección clásica, realizamos la validación cruzada:




```
estadisticas.modelos.final # Mostramos la media y desviacion tipica obtenida en la validacion cruzada
```

```
##     modelo     media         sd
## 1 Modelo 1 0.7143398 0.01387191
## 2 Modelo 2 0.7135692 0.01379937
## 3 Modelo 3 0.7143398 0.01387191
## 4 Modelo 4 0.7135692 0.01379937
## 5 Modelo 5 0.7254781 0.01307943
## 6 Modelo 6 0.7223483 0.01331826
```

Analizando los resultados obtenidos en la selección aleatoria y en la validacion cruzada, los mejores modelos obtenidos han sido el número 2 y el número 6 en términos de desviación de típica (0.01379937 y 0.01331826, respectivamente). No obstante, pese a que la diferencia entre el R2 de entrenamiento y prueba sea ligeramente mayor en el modelo 6 (0.002), los valores tanto de AIC como SBC son menores en comparación con el modelo 2. De hecho, si lo comparamos con el criterio AIC original obtenido en el modelo 1.2 (48908), en el modelo 6 solo se ve aumentado en tan solo 77 puntos, en comparación con los más de 100 que supondría emplear el modelo número 2. Bien es cierto que la anchura de la caja es mayor en el sexto modelo, aunque (dada la escala del gráfico) por una diferencia de apenas décimas o incluso centésimas.

Una vez escogido el modelo 6, realizamos una selección de variables aleatoria, comparando los resultados con el modelo anterior:

```
## Train:  0.71448 ; Test:  0.7135041 ; Dif. (Train-Test):  0.0009759169 ; AIC: 49206.75 ; SBC:  49362.66
## Numero de variables:  22
## Train:  0.7172158 ; Test:  0.7138813 ; Dif. (Train-Test):  0.00333455 ; AIC: 49156.2 ; SBC:  49352.79
## Numero de variables:  28
## Train:  0.7187673 ; Test:  0.7158142 ; Dif. (Train-Test):  0.002953111 ; AIC: 49118.46 ; SBC:  49308.27
## Numero de variables:  27
```

```
##     modelo     media         sd
## 1 Modelo 1 0.7223483 0.01331826
```

```
## 2 Modelo 2 0.7122686 0.01390150
## 3 Modelo 3 0.7141109 0.01394756
## 4 Modelo 4 0.7162326 0.01381079
```

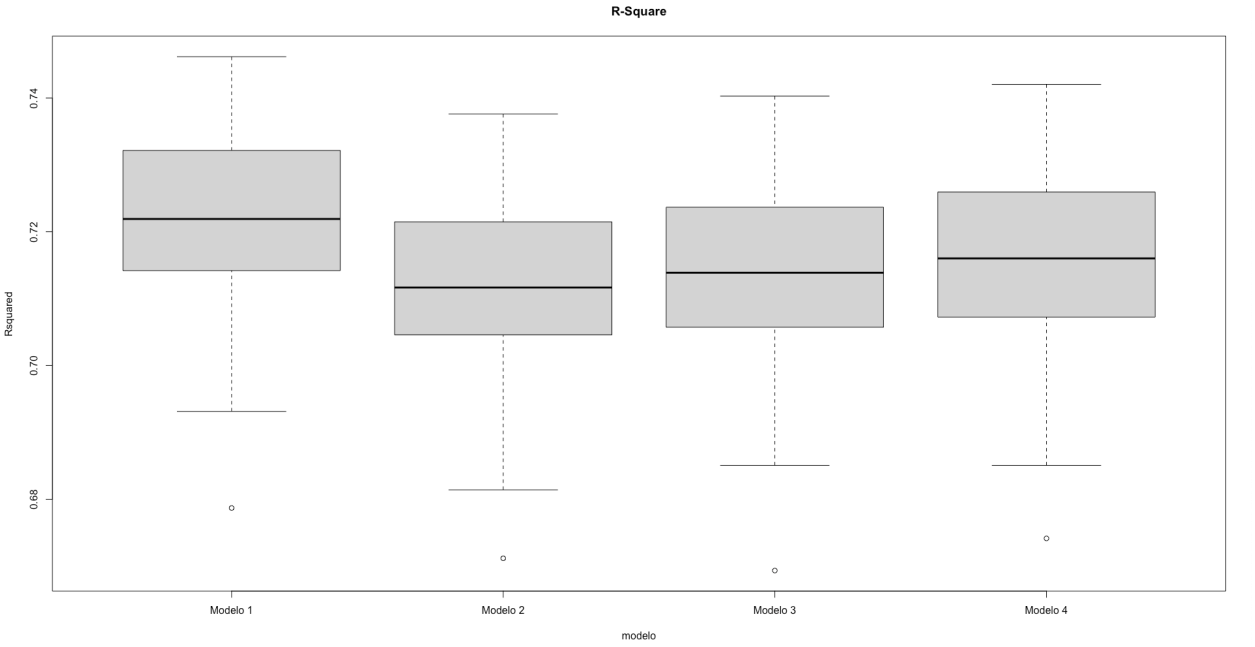


Figure 1: Validación cruzada en el modelo de selección aleatoria + modelo 6

Nuevamente, tanto los valores de AIC como BIC en los modelos aleatorios son mayores con respecto al modelo 6 (incluso el criterio SBC que penalizaba especialmente el número de parámetros). Por otro lado, las desviaciones típicas obtenidas (aunque por una diferencia de tan solo unas centésimas) son superiores a las del modelo 6. **En conclusión, aunque el principio de parsimonia nos lleve a elegir segundo modelo aleatorio (Modelo 3 en el diagrama anterior), pues presenta el mejor resultado de todos los modelos aleatorios ademas de proporcionar la explicación más sencilla al problema, el modelo 6 obtenido en la selección clásica refleja un mejor resultado en todos los sentidos, tanto en AIC, SBC como en desviación típica**. Por tanto, **elegimos como ganador al modelo 6**. A continuación, evaluamos su resultado:

```
# Evaluamos las estadisticas del modelo ganador
mostrar.estadisticas(modelo.final, data_train, data_test, "lm", "varObjCont")
```

```
## Train:  0.7249686 ; Test:  0.7226587 ; Dif. (Train-Test):  0.002309877 ; AIC: 48965.62 ; SBC:  49128.32
## Numero de variables:  23
```

```
##
## Call:
## lm(formula = as.formula(formula.final), data = data_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -54.313  -6.019   0.125   6.424  45.092
##
## Coefficients:
##                                          Estimate Std. Error t value
## (Intercept)                              55.63708    2.84104  19.583
## CCAAAN_NA                                 9.77418    4.02551   2.428
## CCAAAR_CM                                 2.58647    2.83749   0.912
```

```
## CCAACAT_PV                                       18.27406    3.87926   4.711
## CCAACV_EX_AS_BA_CA                                8.75141    3.35333   2.610
## CCAAMA_CA_RI_CE_ME_MU_GA_CM                      14.40237    3.87261   3.719
## Age_19_65_pct                                    -0.20244    0.02234  -9.062
## SameComAutonPtge                                  0.20308    0.02802   7.249
## prop_missings                                     0.06783    0.01794   3.782
## logxForeignersPtge                                0.00831    0.07728   0.108
## logxIndustryUnemploymentPtge                     -0.21004    0.03872  -5.424
## logxServicesUnemploymentPtge                     -0.35826    0.05790  -6.187
## logxtotalEmpresas                                -0.37346    0.05932  -6.296
## CCAAAN_NA:SameComAutonPtge                       -0.35421    0.04786  -7.401
## CCAAAR_CM:SameComAutonPtge                       -0.12588    0.03472  -3.626
## CCAACAT_PV:SameComAutonPtge                      -0.80963    0.04769 -16.977
## CCAACV_EX_AS_BA_CA:SameComAutonPtge              -0.26659    0.04102  -6.498
## CCAAMA_CA_RI_CE_ME_MU_GA_CM:SameComAutonPtge     -0.22260    0.04756  -4.680
## CCAAAN_NA:logxForeignersPtge                      1.59805    0.23560   6.783
## CCAAAR_CM:logxForeignersPtge                      0.41549    0.12411   3.348
## CCAACAT_PV:logxForeignersPtge                    -0.40383    0.31135  -1.297
## CCAACV_EX_AS_BA_CA:logxForeignersPtge             0.62446    0.22571   2.767
## CCAAMA_CA_RI_CE_ME_MU_GA_CM:logxForeignersPtge    0.26749    0.24675   1.084
##                                                 Pr(>|t|)
## (Intercept)                                      < 2e-16 ***
## CCAAAN_NA                                       0.015207 *
## CCAAAR_CM                                       0.362047
## CCAACAT_PV                                      2.52e-06 ***
## CCAACV_EX_AS_BA_CA                              0.009081 **
## CCAAMA_CA_RI_CE_ME_MU_GA_CM                     0.000202 ***
## Age_19_65_pct                                    < 2e-16 ***
## SameComAutonPtge                                4.70e-13 ***
## prop_missings                                   0.000157 ***
## logxForeignersPtge                              0.914373
## logxIndustryUnemploymentPtge                    6.04e-08 ***
## logxServicesUnemploymentPtge                    6.50e-10 ***
## logxtotalEmpresas                               3.25e-10 ***
## CCAAAN_NA:SameComAutonPtge                      1.53e-13 ***
## CCAAAR_CM:SameComAutonPtge                      0.000290 ***
## CCAACAT_PV:SameComAutonPtge                      < 2e-16 ***
## CCAACV_EX_AS_BA_CA:SameComAutonPtge             8.73e-11 ***
## CCAAMA_CA_RI_CE_ME_MU_GA_CM:SameComAutonPtge    2.92e-06 ***
## CCAAAN_NA:logxForeignersPtge                    1.28e-11 ***
## CCAAAR_CM:logxForeignersPtge                    0.000819 ***
## CCAACAT_PV:logxForeignersPtge                   0.194661
## CCAACV_EX_AS_BA_CA:logxForeignersPtge           0.005681 **
## CCAAMA_CA_RI_CE_ME_MU_GA_CM:logxForeignersPtge 0.278383
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.47 on 6473 degrees of freedom
## Multiple R-squared:  0.725,  Adjusted R-squared:  0.724
## F-statistic: 775.6 on 22 and 6473 DF,  p-value: < 2.2e-16
```

A modo de ejemplo, interpretamos los coeficientes de dos variables incluidas en el modelo:

1. **CCAAAN__NA** (Andalucía y Navarra): 9.77. Es decir, **el porcentaje de votos a la derecha aumenta en un 9.77 % si la Comunidad Autónoma a la que pertenece el municipio es Andalucía o Navarra, con respecto a la CCAA de referencia (Castilla y León)**.

2. **logxServicesUnemploymentPtge**: -0.35. Es decir, **por cada incremento unitario en el porcentaje de votos a la Derecha, el porcentaje de parados en el sector Servicios (en escala**

logarítimica) se ve reducido en -0.35 unidades

En última instancia, realizamos un análisis de la importancia de las variables del modelo final:

```
## lm(formula = as.formula(formula.final), data = data_train)
##
## Coefficients
##                                    SSR df pEta-sqr dR-sqr
## (Intercept)                  42000.9894  1   0.0559     NA
## CCAA                          3934.9694  5   0.0055 0.0015
## Age_19_65_pct                 8994.1895  1   0.0125 0.0035
## SameComAutonPtge              5754.5988  1   0.0081 0.0022
## prop_missings                 1566.1984  1   0.0022 0.0006
## logxForeignersPtge               1.2663  1   0.0000 0.0000
## logxIndustryUnemploymentPtge  3222.0221  1   0.0045 0.0013
## logxServicesUnemploymentPtge  4192.3830  1   0.0059 0.0016
## logxtotalEmpresas             4341.2971  1   0.0061 0.0017
## CCAA:SameComAutonPtge        35750.5624  5   0.0480 0.0139
## CCAA:logxForeignersPtge       6352.7036  5   0.0089 0.0025
##
## Sum of squared errors (SSE): 708912.0
## Sum of squared total  (SST): 2577567.7
```

En este caso, nos encontramos con dos variables que apenas aportan valor al R2: **logxForeignersPtge** y **prop_missings**. No obstante, la eliminación de alguna de ellas no aporta mejoría al modelo. En el caso de **logxForeignersPtge**, al eliminarlo supondría que el efecto del porcentaje de extranjeros sobre el porcentaje de votos a la derecha dependiense únicamente de las interacciones de dicha variable con las CCAA. Sin embargo, la interacción entre Castilla y León (la cual se empleaba previamente como categoría de referencia), tampoco aporta información al modelo. Por otro lado, aunque el campo **prop_missings** suponga una pérdida de tan solo el 0.006 en el R2, al eliminarlo tanto el valor de AIC como SBC aumentan con respecto al modelo anterior:

```
# AIC anterior: 48965.62 ---- SBC anterior: 49128.32
Train:  0.724361 ; Test:  0.7213045 ; Dif. (Train-Test):  0.003056495 ; AIC: 48977.96 ; SBC:  49133.87
Numero de variables:  22
```

Además, el p-valor obtenido en el *summary* asegura la importancia de la variable al 95 % de confianza, por lo que tampoco la eliminamos.