Minería de Datos y Modelización Predictiva (I)

Fernández Hernández, Alberto. 54003003S

12/01/2021

Contents

1. Depuración de los datos **2**
1.1 Introducción al objetivo del problema y las variables implicadas . . . 2
1.2 Valores erróneos o no declarados . . . 3
1.3 Análisis de valores atípicos . . . 3
1.4 Análisis de valores missings (NA). Imputaciones . . . 4
1.4 Relaciones con las variables input y objetivo . . . 8
1.5 Transformaciones de variables y relaciones con las variables objetivo . . . 9

2. Construcción del modelo de regresión lineal **10**
2.1 Selección de variables clásica . . . 11
2.2 Selección de variables aleatoria . . . 12
2.3 Selección y justificación del modelo ganador . . . 12
2.4 Interpretación de los coeficientes de dos variables . . . 15

3. Construcción del modelo de regresión logística **15**
3.1 Selección de variables clásica . . . 16
3.2 Selección de variables aleatoria . . . 18
3.3 Selección y justificación del modelo ganador . . . 18
3.4 Selección del punto de corte óptimo . . . 20
3.5 Interpretación de los coeficientes de dos variables . . . 21

1. Depuración de los datos

1.1 Introducción al objetivo del problema y las variables implicadas

El objetivo principal del problema consiste en **obtener un modelo tanto de regresión lineal como de regresión logística que permita calcular no solo el porcentaje de votos a la derecha en un municipio (Dcha_Pct), sino además predecir si en un municipio habrá una mayoría o no de votos a la derecha (maximizando tanto verdaderos positivos como negativos)**.

Inicialmente (y una vez eliminadas el resto de variables objetivo) nos encontramos ante un conjunto de datos con la información demográfica de los diferentes municipios en España así como sus últimos resultados electorales. En primer lugar, y antes de analizar las variables independientes, debemos **recategorizar las variables cualitativas como factor**, dado que el formato establecido por defecto es numérico o cadena de caracteres. Según la documentación adjunta, existen un total de 4 variables categóricas, incluyendo la variable objetivo cualitativa: Derecha, CodigoProvincia, CCAA, ActividadPpal y Densidad, dado que contienen un número limitado de valores únicos:

```
# c(2,3,7,29,33) -> (CodigoProvincia, CCAA, Derecha, ActividadPpal, densidad)
datos[,c(2,3,7,29,33)] <- lapply(datos[,c(2,3,7,29,33)], 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. Numero columnas:  34
```

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 el campo identificador
```

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 el campo o lo eliminamos? 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 deben ser agrupadas. Además, nos encontramos con un segundo problema: **el campo CCAA y CodigoProvincia están muy correlacionados según la V de Cramer**:

```
sapply(datos[, c("CodigoProvincia")],function(x) Vcramer(x,datos$CCAA)) # Correlacion perfecta (1)
```

```
## CodigoProvincia
##               1
```

¿Cuál debemos eliminar? En primera instancia, deberíamos descartar aquella variable que esté menos relacionada con nuestras variables objetivo, según la V de Cramer:

```
sapply(datos[, c("CodigoProvincia", "CCAA")],function(x) Vcramer(x,varObjCont))
```

```
## CodigoProvincia            CCAA
##       0.5323588       0.5133780
```

```
sapply(datos[, c("CCAA", "CodigoProvincia")],function(x) Vcramer(x,varObjBin))
```

```
##            CCAA CodigoProvincia
##       0.7003718       0.7119410
```

Como primeros resultados, la V de Cramer obtenida nos indica que el código de la provincia está mejor relacionada con las variables objetivo. Sin embargo, debemos recordar el número de categorías de cada variable:

```
## Categorias CodigoProvincia:  52 ; CCAA:  19
```

Es decir, con 52 - 19 = 33 categorías menos, la diferencia entre ambas relaciones es de apenas 0.011 en la variable objetivo binaria y de 0.018 en la variable objetivo continua, con muchas menos categorías, por lo que no parece ser tan necesario conocer de qué provincia proviene el municipio, sino que con la CCAA parece ser suficiente. Por otro lado, muchas de las categorías en CódigoProvincia podrían estar muy poco representadas. A modo de ejemplo, de las 52 provincias, 18 de ellas tienen menos de 100 valores en el conjunto de datos, lo que podría suponer no solo recategorizarlas, sino además de ser un proceso computacionalmente más costoso de cara a la elaboración de los modelos (especialmente en la regresión logística):

```r
sum(freq(datos$CodigoProvincia)$`n` < 100)
```

```
## [1] 18
```

Por otro lado, ¿Y si lo consideramos como variable numérica? ¿Mejora la V de Cramer?

```
## VarObjCont:  0.14539 ; VarObjBin:  0.2371896
```

Tampoco parece mejorar. Por tanto, dado que la diferencia V Cramer entre ambas variables no es tan significativa pese aumentar el número de categorías, **elegimos la CCAA, por lo que eliminamos CodigoProvincia**. 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:

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

1.2 Valores erróneos o no declarados

A continuación, procedemos a eliminar aquellos valores no declarados en las variables, así como posibles 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
```

```r
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 mayores a 100):

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

```r
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
```

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

3. Valores a 99.999 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
```

```r
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
```

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

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

1.3 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 una elevada asimetría, **en indicios de la presencia de valores atípicos** (algo lógico si obervamos las desviaciones típicas obtenidas, donde en algunas variables como en el caso de *Population* presentan valores muy extremos, del orden de 46.000). Por ello, comenzamos analizando el porcentaje máximo de valores atipicos en nuestro conjunto de datos (top 5):

```
##           Servicios       totalEmpresas          Population ComercTTEHosteleria
##           11.873383           10.506220            9.927331            9.853430
##             Pob2010
##            9.754896
```

En este caso, el maximo porcentaje corresponde con el campo *Servicios*, con un 11.87 %, además de que las variables con mayor porcentaje **corresponden con aquellas de elevada asimetría**. No obstante, ¿Es tan elevado el porcentaje de atípicos en cada columna? Veamos el porcentaje de *outliers* mediante la función *summary*:

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##  0.00000  0.01232  0.65279  3.57739  8.86809 11.87338
```

Si nos fijamos en el tercer cuartil, podemos comprobar que no todas las columnas presentan un alto porcentaje de atípicos. De hecho, el 75 % de las columnas no supera el 10 % de atípicos (menos 800 filas en un conjunto de datos de más de 8.000), por lo que dada la proporción podemos considerar dichos valores como atípicos, por lo que los recategorizamos como *missing*:

```
## Total valores missing:  10064
```

1.4 Análisis de valores missings (NA). Imputaciones

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

```
## [1] 12.63702
```

```
##   Por.observacion Por.variable
## 1            37.5        12.64
```

Aparentemente, mientras que el porcentaje de *missings* por variable es del 12.64 %, por observaciones detectamos un mayor número (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 varias filas (de hecho, solo el 25 % presenta un porcentaje de *missings* superior al 3 %, con una media muy pequeña en comparación con el valor máximo, lo que indica que se tratan de casos atípicos). Por otro lado, la pérdida de información en ambos casos no supera el 50 %, por lo que en lugar de eliminar las filas o columnas podemos imputarlos. No obstante, existen determinados campos que pueden ser imputados manualmente sin necesidad de emplear una media, mediana o de forma aleatoria:

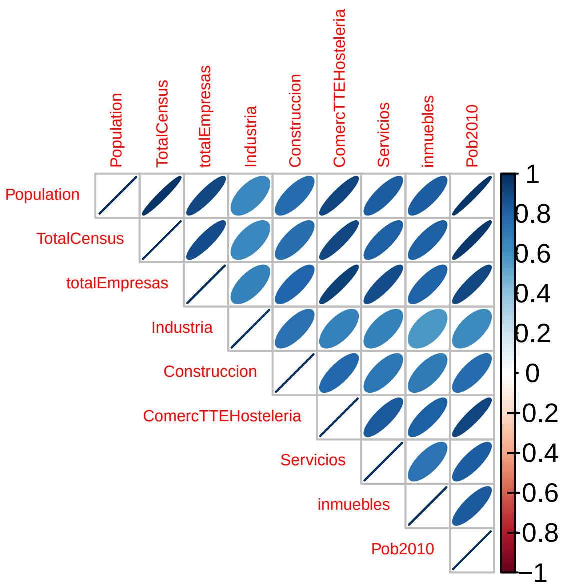
1. *Age_19_65_pct*, cuyo porcentaje de edad puede calcularse a partir de la suma de *Age_under19_Ptge* y *Age_over65_pct* menos el 100 %:

```
x["Age_19_65_pct"] <- 100 - (as.numeric(x["Age_under19_Ptge"]) + as.numeric(x["Age_over65_pct"]))
# Ejemplo demostrativo de que 100 - Age_under19_Ptge + Age_over65_pct = Age_19_65_pct
```

```
##   Age_19_65_pct X100.Age_under_19.Age_over65
## 1        55.059                       55.060
## 2        56.643                       56.641
```

```
## 3         54.834                        54.833
## 4         60.098                        60.096
## 5         59.391                        59.389
```

2. *totalEmpresas* ¿Podría calcularse a partir de la suma del número de empresas de cada sector? *Industria*, *Construcción*, *ComercioTTEHostelería* y *Servicios*. Una primera prueba para comprobar si el campo *totalEmpresas* es la suma de cada columna podría ser analizando la matriz de correlación de valores *missings*: **Si falta cualquier sector, totalEmpresas tampoco debería aparecer al no poder calcularse**:



Efectivamente, **detectamos una correlación entre los valores *missing* de *totalEmpresas* con cada sector**. De hecho, no solo existe correlación entre *totalEmpresas*, sino incluso entre cada sector. A modo de ejemplo, si el número de empresas dedicadas al comercio no aparece, el número de empresas dedicadas a la construcción tampoco. Incluso si la población no aparece, tampoco suelen aparecer el número de empresas, el censo total e incluso tampoco la población registrada en el año 2010 (No se tratan de valores *missing* aleatorios, siguen un patrón, un posible indicio de colinealidad). Por tanto, el campo *totalEmpresas* puede calcularse a partir de la suma de cada sector.

Como última prueba, realicemos una comprobación manual, sumando cada columna para comprobar si coincide con *totalEmpresas*:

```
x["totalEmpresas"] <- as.numeric(x["Industria"]) + as.numeric(x["Construccion"]) +
                      as.numeric(x["ComercTTEHosteleria"]) + as.numeric(x["Servicios"])
# Ejemplo demostrativo de la suma de las empresas de cada sector, en funcion de ActividadPpal
```

```
##                ActividadPpal
## Coincide_Suma. ComercTTEHosteleria Construccion Industria Otro Servicios
##            NO                   0            0         0 3712         0
##            SI                1700           11         9 1215       195
```

Por lo general, cuando la actividad principal es *Otro*, la suma de cada columna no suele coincidir con *totalEmpresas* (al ser predominante otro tipo de Actividad, el resto de columnas valen 0). No obstante, de los valores *missing* de *totalEmpresas*, sólo existen 5 filas con la actividad principal a *Otro*, por lo que podemos realizar el cálculo manual sin problema alguno:

```
##
## ComercTTEHosteleria        Construccion           Industria                Otro
##                 527                   0                   0                   5
##           Servicios
##                 326
```

Sin embargo, dado que existen valores *missing* tanto de *Industria*, *Construcción*, *ComercTTEHosteleria* como *Servicios*, realizaremos el cálculo manual una vez imputados el resto de campos:

```
## Num. Missings Industria: 898 ; Construccion: 863 ; ComercTTEHosteleria: 809 ; Servicios: 1026
```

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

```
ifelse(proporcion < 1, densidad <- "MuyBaja", ifelse(proporcion >= 1 & proporcion <= 5,
      densidad <- "Baja", densidad <- "Alta"))
# Ejemplo demostrativo del cociente entre Population y SUPERFICIE a traves de la funcion table
```

```
##          Cociente_Pob_SUP
## Densidad  Entre1Y5 Mayor5 Menor1
##   Alta           0    140      0
##   Baja         755      0      0
##   MuyBaja        0      0   6152
```

De nuevo, dado que existen valores *missing* de *Population* y *SUPERFICIE*, realizaremos el cálculo de la Densidad una vez realizada la imputación en ambos campos:

```
## Num. Missings Population: 806 ; SUPERFICIE: 229
```

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 como pudimos comprobar anteriormente, lo que hace que la mediana sea mucho más representativa que la media):

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

```
## 1215  valores missing tras la imputacion aleatoria (NO se han eliminado todos)
```

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

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

Como podemos observar, hemos conseguido reducir el porcentaje de *missings*. A continuación, si imputamos manualmente las tres columnas mencionadas anteriormente conseguimos reducir tanto el numero de *missing* como el porcentaje máximo de atípicos:

```
## 0  valores missing. Columna con mayor % atipicos:  10.08745
```

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[-30, -30] - corr.previa
sum(abs(comparacion.corr) < 0.2) * 100 / (dim(comparacion.corr)[1] * dim(comparacion.corr)[2])
```

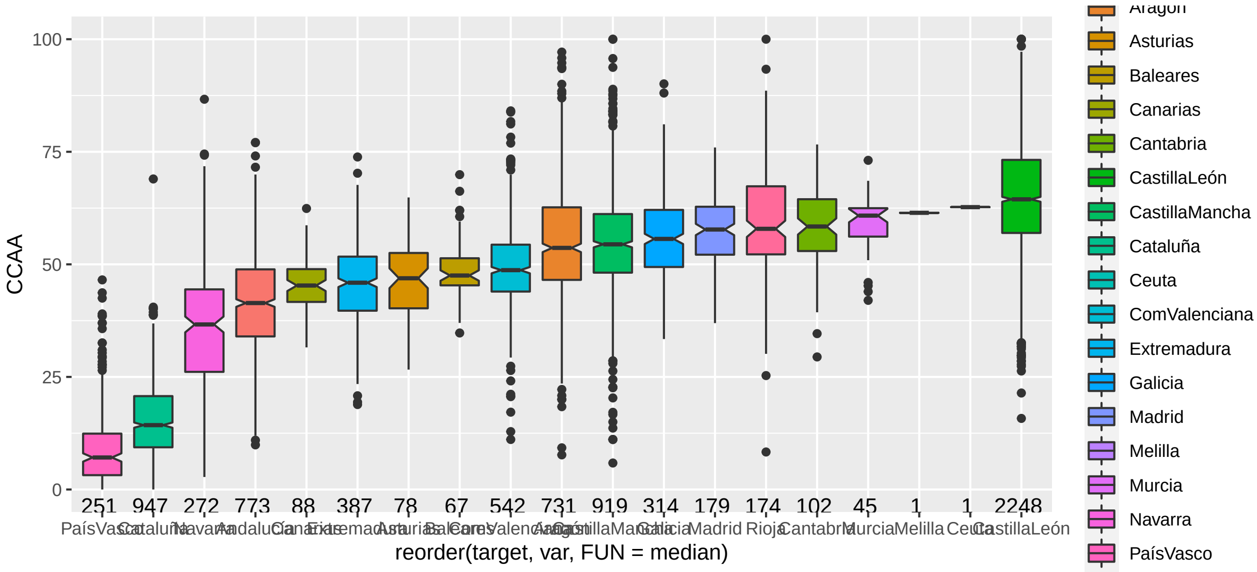
```
## [1] 77.17004
```

Podemos observar que aproximadamente **un 76 % de las correlaciones originales ha variado en menos de 0.2 con respecto a su correlación original**, bastante mayor con respecto a una imputación únicamente con la media o con la mediana.

Tras imputar las variables cuantitativas, 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, disponemos de 19 valores diferentes, algunos de los cuales como Ceuta o Melilla con una única representación, tal y como se muestra a continuación (en la base de cada *boxplot* se encuentra el número de ocurrencias de cada categoría):



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** (PV_CAT); **Navarra y Andalucía** (AN_NA); **ComValenciana, Extremadura, Asturias, Baleares y Canarias** (CV_EX_AS_BA_CA); **Aragón y Castilla la Mancha**; (AR_CM) así como **Galicia, Cantabria, Madrid, La Rioja, Ceuta, Melilla y Murcia'** (MA_CA_RI_CE_ME_MU_GA). De este modo, no sólo conseguiremos concentrar auquellas CCAA con una distribución de votos similar, sino además reducir el número de categorías. 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):

```
##        AA_CL   AN_NA   AR_CM  CAT_PV CV_EX_AS_BA_CA MA_CA_RI_CE_ME_MU_GA
## n     2248.0  1045.0  1650.0  1198.0         1162.0                816.0
## %       27.7    12.9    20.3    14.8           14.3                 10.1
## val%    27.7    12.9    20.3    14.8           14.3                 10.1
```

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, las categorías *Construccion* e *Industria* apenas tienen 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 la categoría con mayor representación: *Otro*, dado que la mediana en las tres categorías es muy similar:

```
summary(estadisticas$data[estadisticas$data$target == "Otro", "variable"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   42.30   56.00   52.66   67.24  100.00
```

```
summary(estadisticas$data[estadisticas$data$target == "Construccion", "variable"])
```

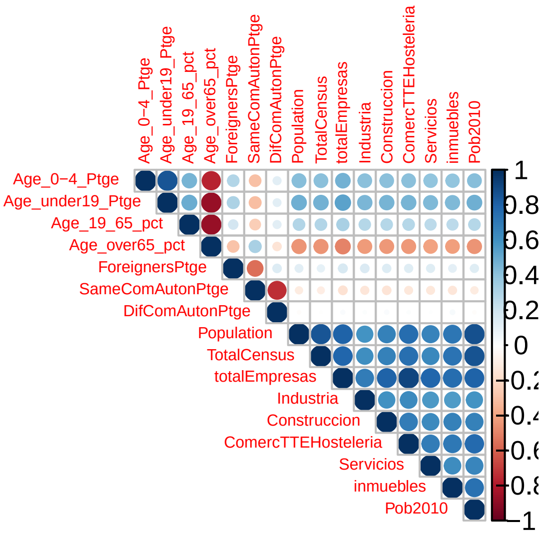
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.386  43.243  56.004  46.927  61.925  70.855
```

```r
summary(estadisticas$data[estadisticas$data$target == "Industria", "variable"])
```

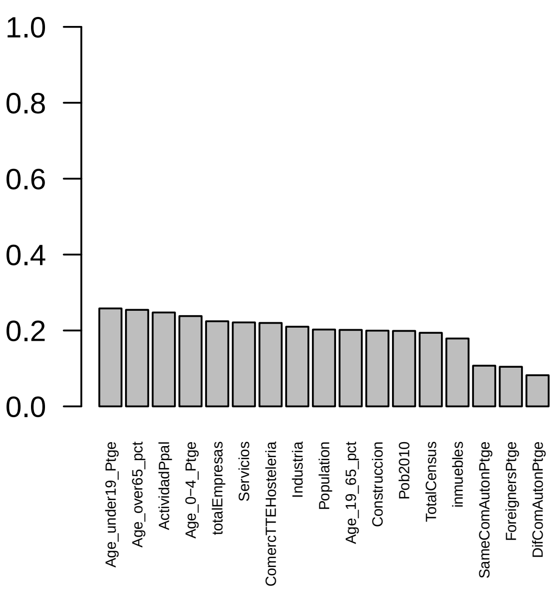
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   6.182  49.708  57.353  50.734  64.276  74.774
```

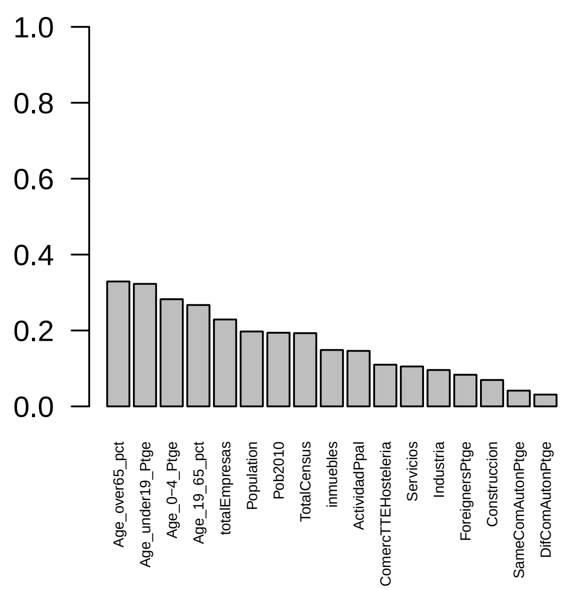
1.4 Relaciones con las variables input y objetivo

Una vez recategorizadas las variables, ¿Cómo están relacionadas las variables *input* con las variables objetivo? O incluso algo más importante ¿Existe colinealidad entre las variables? Para responder a esta última pregunta, debemos analizar el siguiente subconjunto de la matriz de correlación:



Analizando el gráfico, debemos destacar tres grandes grupos de correlación. En primer lugar, **las edades**, donde cada porcentaje puede llegar a obtenerse (como pudimos observar con *Age_19_65_pct*) a partir del resto de edades, es decir, **son complementarios**. Por otro lado, **el porcentaje de personas que residen en la misma o diferente CCAA**, e incluso entre dichos porcentaje y el porcentaje de extranejos. Como último bloque nos encontramos no solo con *totalEmpresas* y el resto de sectores (dado que son campos complementarios), sino además con *Population*, *TotalCensus*, *inmuebles* y *Pob2010*, campos en los que pudimos detectar una elevada correlación entre valores *missing*. Por tanto, ¿Debemos eliminar algún campo? La respuesta es si, pero con cierto cuidado ya que solo podemos eliminar uno de cada grupo complementario (si eliminamos más de uno no podríamos volver a calcularlo). Para analizar qué campos podemos o no eliminar, realicemos la V de Cramer para ambas variables objetivo:





En ambas variables obejtivo, de los campos de edad podemos eliminar *Age_19_65_pct* con menor correlación; del porcentaje de residencia en la misma o diferente CCAA podemos descartar *DifComAutonPtge*. Por último, ¿Qué podemos hacer con el bloque de *totalEmpresas*? En relación con la variable objetivo continua, la V de Cramer nos indica que **todos las variables correlacionadas presentan aproximadamente la misma importancia con respecto a la variable objetivo**, por lo que no podemos descartar ninguna hasta tener el modelo final (puede haber alguna variable que aumente su importancia más adelante o incluso mediante interacciones). No obstante, el número de inmuebles presenta una menor correlación con respecto al resto de campos, por lo que podemos eliminarla. En relación con la variable binaria, podríamos eliminar el campo *Construcción*, dado que su importancia es menor con respecto al resto, además de que podría calcularse a partir del resto de empresas, tal y como hicimos con la imputación manual de *totalEmpresas*. Con respecto al resto de campos, insisto, no podemos eliminarlos aún, dado que pueden resultar de utilidad de cara a ambos modelos.

1.5 Transformaciones de variables y relaciones con las variables objetivo

Tras eliminar las variables menos relevantes, debemos realizar las trasnformaciones de las variables continuas con el objetivo de que el modelo de predicción funcione mejor o **pueda plasmar la verdadera relación con las variables objetivo**:

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

La cuestión es ¿todas las transformaciones son significativas? ¿Aportan mejoría a los modelos? Dado que computacionalmente sería muy costoso no solo trabajar con las variables originales, sino además con sus transformadas (especialmente en el modelo de regresión logística), **filtraremos únicamente aquellas transformadas que mejoren en gran medida a la variable original**. De hecho, no tendría mucho sentido mantener en un modelo tanto las variables originales como sus transformadas, por ejemplo *foreignersPtge* y *logxForeignersPtge* en el mismo modelo. Por ello, comenzando con la variable objetivo cuantitativa filtraremos aquellas transformaciones cuya correlación con respecto a la variable objetivo mejore en más de 0.1 con respecto a la variable original, dado que (como podemos observar en el tercer cuartil del siguiente *summary*), sólo un el 25 % de las variables originales ve mejorado su correlación en más de 0.1, por lo que en el resto de variables la mejoría es prácticamente nula:

```
correlaciones  <- round(abs(cor(Filter(is.numeric, input_cont), use="pairwise", method="pearson"))[1,29:55]
summary(correlaciones)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00000 0.00500 0.07000 0.06926 0.11000 0.20000
```

```
# Filtramos unicamente las transformadas que mejoren en mas de 0.1
input_cont <- input_cont[, !colnames(input_cont) %in% names(correlaciones[correlaciones < 0.1])]
input_cont <- input_cont[, c(-3,-4,-9,-12,-14,-15,-17,-19,-25)]
# Todas las transformadas significativas emplean escalas logaritmicas
names(correlaciones[correlaciones >= 0.1])
```

```
## [1] "logxPopulation"                    "logxTotalCensus"
## [3] "logxForeignersPtge"                "logxUnemployLess25_Ptge"
## [5] "logxUnemployMore40_Ptge"           "logxAgricultureUnemploymentPtge"
## [7] "logxConstructionUnemploymentPtge" "logxtotalEmpresas"
## [9] "logxPob2010"
```

En relación con la variable objetivo binaria, para estudiar la importancia de las variables empleamos un criterio mucho más preciso que la V de Cramer: el criterio del **Valor de la Información**, una medida que permite analizar la influenza o poder predictivo que presenta una variable sobre otra dicotómica, por lo que cuanto mayor sea su valor de información o IV (generalmente a partir de 0.1), se dice que su poder predictivo es fuerte o influyente. En este caso, al igual que en la matriz de correlación restaremos los valores de información tanto de las variables transformadas como originales con el objetivo de analizar si la mejora es o no significativa:

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

```
summary(sapply(salida.woe[c(31:57)], function(x) x$total_iv[1]) -
          sapply(salida.woe[c(2:21,23:25,27:30)], function(x) x$total_iv[1]))
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -0.0153878 -0.0033109 -0.0008152 -0.0005668  0.0014226  0.0201268
```

Como podemos observar a partir de la salida anterior, un 75 % las variables transformadas **ve mejorado su valor de información en 0.001 o menos**, e incluso empeora ligeramente en algunos casos. Por otro lado, el aporte máximo al valor de información ha sido 0.02, un valor muy poco significativo. Si a ello le añadimos que las transformadas son de tipo "x", es decir, su valor multiplicado por 1.0001 (funcionesRosa.R), descartamos por tanto las variables transformadas del modelo de regresión logístico:

```
names(input_bin)[40:43] # Ejemplo del tipo de transformacion
```

```
## [1] "xSameComAutonDiffProvPtge" "xUnemployLess25_Ptge"
## [3] "xUnemploy25_40_Ptge"       "xUnemployMore40_Ptge"
```

Finalmente, una vez completado el proceso de depuración ya tenemos nuestros conjuntos de datos preparados para elaborar los modelos de regresión lineal y logísticos:

```
## Numero de columnas finales en input_cont:  31 ; input_bin:  31
```

2. Construcción del modelo de regresión lineal

Comenzamos con el modelo de regresión lineal. Inicialmente, una vez divididos el conjunto de datos en entrenamiento y prueba (80 %, 20 % respectivamente), **realizaremos una primera regresión con todas y cada una de las variables del modelo, incluidas todas las posibles interacciones**. De este modo, aunque no sea el modelo definitivo podremos filtrar aquellas variables más relevantes de cara a facilitar el proceso de selección clásica en lugar de ejecutar directamente la selección con todos los posibles parámetros:

```
formInt<-formulaInteracciones(input_cont,1)
modelo1<-lm(formInt,data=data_train)
# Funcion que muestra tanto el AIC - SBC - R2 train y test (y su diferencia) - Num. parametros
mostrar.estadisticas(modelo1, data_train, data_test, "lm", "varObjCont")
```

```
## Train:  0.7529327 ; Test:  0.6913446 ; Dif. (Train-Test):  0.0615881 ; AIC: 48831.09 ; SBC:  50898.67
## Numero de variables:  304
```

Analizando las estadísticas obtenidas, observamos que el R2 obtenido en el conjunto de prueba es significativamente menor que en el conjunto de entrenamiento (diferencia de 0.06 entre ambos), lo que implica un claro sobreajuste en el modelo y, como consecuencia, **un exceso de parámetros**. Con el objetivo de mejorar el modelo, analicemos la importancia de cada una de las variables con respecto al modelo inicial, mediante la función *modelEffectSizes*:

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

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
## 0.0000007 0.0000380 0.0000959 0.0002327 0.0002187 0.0065731         1
```

Analizando la salida, debemos destacar el tercer cuartil: de todas las variables del modelo, **el 75 % aportan 0.0002 o menos al R2 final, un valor muy pequeño en comparación con otras variables donde el aporte es de 0.006 (valor máximo)**. Es decir, existe un contraste entre variables poco significativas y variables muy significativas. Por ello, analicemos las variables más atípicas, es decir, las que aportan mayormente al R2:

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

```
##      [,1]                             [,2]
## [1,] "Age_under19_Ptge"               "CCAA:PersonasInmueble"
## [2,] "CCAA:Age_0.4_Ptge"              "CCAA:Explotaciones"
## [3,] "CCAA:Age_under19_Ptge"          "CCAA:logxForeignersPtge"
## [4,] "CCAA:SameComAutonPtge"          "CCAA:logxAgricultureUnemploymentPtge"
## [5,] "CCAA:SameComAutonDiffProvPtge" "CCAA:logxtotalEmpresas"
## [6,] "CCAA:ActividadPpal"             "Age_under19_Ptge:Densidad"
## [7,] "CCAA:PobChange_pct"             ""
```

Salvo el campo *Age_under19_Ptge*, el resto de interacciones parecen ser las más significativas en el modelo original, interactuando especialmente con la Comunidad Autónoma desde el porcentaje de menores de edad (under_19 y Age_0_4) hasta el número de extranjeros o residentes en la misma CCAA o en diferente provincia. Por tanto, de cara a un segundo modelo mantendremos dichas interacciones además de las columnas originales, ya que puede ocurrir que alguna variable sea más significativa sin tener que interactuar con otra:

```
mostrar.estadisticas(modelo1.2, data_train, data_test, "lm", "varObjCont")
```

```
## Train:  0.7391653 ; Test:  0.7281157 ; Dif. (Train-Test):  0.01104957 ; AIC: 48773.34 ; SBC:  49451.24
## Numero de variables:  99
```

Reduciendo el número de parámetros de 304 a 99, el modelo mejora prácticamente en todos los sentidos, tanto un AIC como SBC más bajos, además de recortar la diferencia entre ambos R2 (mejorando en el caso del conjunto de prueba de 0.69 a 0.72).

2.1 Selección de variables clásica

No obstante, el modelo continua teniendo demasiados parámetros, por lo que realizamos una selección clásica empleando este último modelo, mediante los criterios AIC-both, SBC-both, AIC-forward, SBC-forward, AIC-backward y SBC-backward, devolviendo sus resultados en una tabla como sigue a continuación:

```
##              R.2.train  R.2.test  Diferencia      AIC      SBC N.Parametros
## AIC-both     0.7315348 0.7239069 0.007627932 48872.65 49252.27           55
## SBC-both     0.7259493 0.7226639 0.003285443 48950.42 49140.23           27
## AIC-forward  0.7315348 0.7239069 0.007627932 48872.65 49252.27           55
## SBC-forward  0.7264527 0.7223404 0.004112256 48946.47 49163.40           31
## AIC-backward 0.7385320 0.7273508 0.011181250 48757.10 49326.53           83
## SBC-backward 0.7287939 0.7246366 0.004157332 48888.64 49098.78           30
```

Analizando la tabla resultante, dado su menor número de parámetros quisiera destacar tanto el modelo 2 como el modelo 6, ya que el resto no ha disminuido lo suficiente en cuanto al número de variables se refiere. En contraste, la diferencia entre el R2 train-test y su menor número de variables da una ligera ventaja al modelo 2, aunque el modelo 6 no solo mejora en cuanto a AIC se refiere, sino incluso que el criterio SBC (que penaliza el número de parámetros), da una mayor ventaja al modelo 6 (48.888 y 49.098). Para confirmar el mejor modelo clásico, realizamos una validación cruzada **con un total de 20 repeticiones, empleando 5 grupos**:

```
##     modelo     media         sd
## 1 Modelo 1 0.7256984 0.01350954
## 2 Modelo 2 0.7229657 0.01358164
## 3 Modelo 3 0.7256984 0.01350954
## 4 Modelo 4 0.7227941 0.01369763
## 5 Modelo 5 0.7300502 0.01319145
## 6 Modelo 6 0.7259522 0.01319900
```

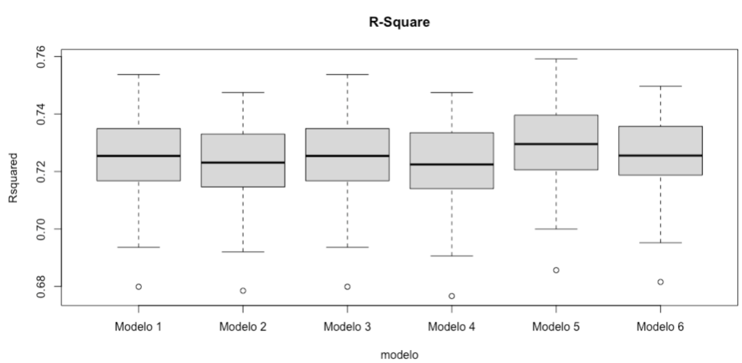


Figure 1: Validación cruzada en el modelo de selección clásica

Los resultados obtenidos en la validación cruzada arrojan tanto una menor desviación típica (0.0135 frente 0.0131) como una mayor media en el R2 del modelo 6 frente al modelo 2 (0.7256 frente 0.7259). De hecho, ¿Qué variables les diferencian?

```
## ¿Que tiene el modelo 2 que no tenga el modelo 6?
##  prop_missings logxAgricultureUnemploymentPtge CCAA:logxForeignersPtge
```

```
## ¿Que tiene el modelo 6 que no tenga el modelo 2?
##  SameComAutonDiffProvPtge Densidad CCAA:SameComAutonDiffProvPtge Age_under19_Ptge:Densidad
```

A la vista de las variables obtenidas, empleando tanto las interacciones *CCAA:SameComAutonDiffProvPtge* y *Age_under19_Ptge:Densidad* parece que el modelo mejora en comparación con las variables del modelo 2. Si observamos además el p-valor de cada modelo vemos que por lo general las variables del modelo 6 son más significativas que las del modelo 2, aún teniendo más variables:

```
## Modelo 2 (el 75 % de las variables tienen un p-valor de 0.005 o menos)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0000000 0.0001028 0.0529605 0.0056376 0.6723438
```

```
## Modelo 6 (el 75 % de las variables tienen un p-valor de 0.001 o menos)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0000000 0.0000017 0.0546477 0.0018766 0.5713570
```

Por tanto, de cara a la selección aleatoria **compararemos los modelos obtenidos con el modelo clásico 6**, ya que obtiene un menor resultado en términos de "bondad media" y de criterio AIC/SBC.

2.2 Selección de variables aleatoria

Como última comparación, realizamos una selección aleatoria **a partir del 70 % de los datos de entrenamiento (por mayor velocidad)**, con el objetivo de comprobar si existe algún otro modelo que mejore el candidato obtenido en la selección clásica. En primer lugar, analizamos las estadísticas de los **tres mejores modelos aleatorios**:

```
## MODELOS ALEATORIOS
```

```
## Modelo aleatorio  1
## Train:  0.7156754 ; Test:  0.7113455 ; Dif. (Train-Test):  0.004329855 ; AIC: 49185.49 ; SBC:  49361.74
## Numero de variables:  25
## Modelo aleatorio  2
## Train:  0.7144549 ; Test:  0.7108354 ; Dif. (Train-Test):  0.003619458 ; AIC: 49209.32 ; SBC:  49372.01
## Numero de variables:  23
## Modelo aleatorio  3
## Train:  0.7140026 ; Test:  0.7111627 ; Dif. (Train-Test):  0.002839889 ; AIC: 49219.6 ; SBC:  49382.29
## Numero de variables:  23
```

En primera instancia, pese a disminuir el número de parámetros, los modelos aleatorios **no mejoran en cuanto a AIC y SBC se refiere**, además de que el valor R2 disminuye ligeramente (de 0.72 en el modelo 6 a 0.71 en los modelos aleatorios) ¿Y en cuánto a la desviación típica?

```
##     modelo     media         sd
## 1 Modelo 1 0.7259522 0.01319900
## 2 Modelo 2 0.7129467 0.01453005
## 3 Modelo 3 0.7122258 0.01389751
## 4 Modelo 4 0.7119924 0.01442047
```

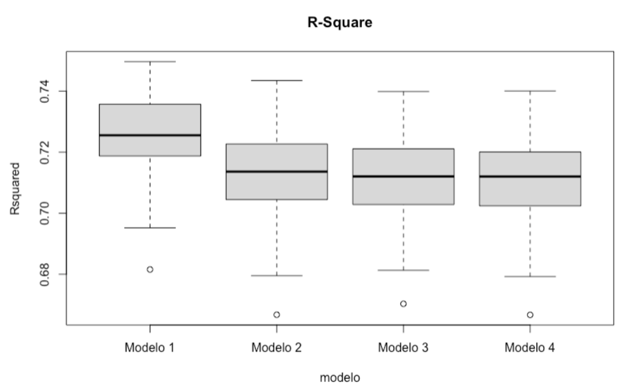


Figure 2: Modelo 6 (Modelo 1 en la imagen) + Validación cruzada en el modelo de selección aleatoria

2.3 Selección y justificación del modelo ganador

Nuevamente, ninguno de los modelos aleatorios consigue mejorar al modelo 6 en términos de media R2 y desviación típica, aunque bien es cierto que el segundo modelo aleatorio presenta una desviación típica muy similar (0.013), aunque con una media menor. Por tanto, de todos los modelos evaluados, **el modelo 6 ofrece un mejor resultado tanto en función del criterio AIC, SBC como en desviación típica**. Sin embargo, no podemos declarar el modelo 6 como ganador sin antes hacernos la siguiente pregunta: ¿Existe correlación en sus variables? Uno de los problemas que pudimos analizar en la fase de depuración fue la elevada correlación que presentan muchas de las variables, tanto las edades, el porcentaje de residencia en la misma CCAA como además del total de empresas. Por tanto, debemos eliminar todas aquellas variables, con menor importancia según la salida en *modelEffectSizes*, que presenten una moderada-alta correlación con el resto de parámetros (superior a 0.4 o inferior a -0.4):

```
Coefficients
                                       SSR df pEta-sqr dR-sqr
(Intercept)                     24398.5990  1   0.0337     NA
CCAA                             4853.6283  5   0.0069 0.0019
Age_under19_Ptge                  404.3579  1   0.0006 0.0002
Age_over65_pct                   7533.5386  1   0.0107 0.0029
SameComAutonPtge                 7534.2054  1   0.0107 0.0029
SameComAutonDiffProvPtge          574.3591  1   0.0008 0.0002
IndustryUnemploymentPtge         2936.2413  1   0.0042 0.0011
ServicesUnemploymentPtge         3648.5332  1   0.0052 0.0014
Construccion                     1012.6068  1   0.0014 0.0004
Densidad                         6505.8942  2   0.0092 0.0025
logxForeignersPtge               2264.8519  1   0.0032 0.0009
logxConstructionUnemploymentPtge 2687.7770  1   0.0038 0.0010
logxtotalEmpresas                4031.4474  1   0.0057 0.0016
CCAA:SameComAutonPtge           42173.0923  5   0.0569 0.0164
CCAA:SameComAutonDiffProvPtge    7399.7126  5   0.0105 0.0029
Age_under19_Ptge:Densidad        7165.5090  2   0.0101 0.0028

Sum of squared errors (SSE): 699052.1
Sum of squared total  (SST): 2577567.7
```

```
                   Age_under19_Ptge Age_over65_pct SameComAutonPtge
Age_under19_Ptge          1.0000000     -0.8644204       -0.3072175
Age_over65_pct           -0.8644204      1.0000000        0.3203304
SameComAutonPtge         -0.3072175      0.3203304        1.0000000
Construccion              0.4644942     -0.4322092       -0.1458669
logxForeignersPtge        0.4498911     -0.4135373       -0.3559378
logxtotalEmpresas         0.6558898     -0.6053973       -0.1489421
                   Construccion logxForeignersPtge logxtotalEmpresas
Age_under19_Ptge      0.4644942          0.4498911         0.6558898
Age_over65_pct       -0.4322092         -0.4135373        -0.6053973
SameComAutonPtge     -0.1458669         -0.3559378        -0.1489421
Construccion          1.0000000          0.2027565         0.4769535
logxForeignersPtge    0.2027565          1.0000000         0.4674308
logxtotalEmpresas     0.4769535          0.4674308         1.0000000
```

Figure 3: Salida modelEffectSizes modelo 6 + Matriz de correlación

1. *Age_under19_Ptge* y *Age_over65_pct*: se ha decidido eliminar *Age_under19_Ptge* (junto con *Age_under19_Ptge:Densidad*) dado que *Age_over65_pct* aporta prácticamente el mismo R2 sin realizar una interacción entre ninguna variable (0.0029 en la columna *dR-sqr*)
2. *Age_over65_pct* con *logxtotalEmpresas*, *Construcción* y *logxForeignersPtge*. En todos los casos anteriores, se elimiman el resto de campos debido a la menor pérdida en el R2 que supone (0.0029 frente a 0.0016, 0.0004 y 0.0009 en la columna *dR-sqr*)

Una vez eliminadas las variables, comparamos los dos modelos (original y eliminando las variables con mayor correlación):

```
## Modelo 6 original

## Train:  0.7287939 ; Test:  0.7246366 ; Dif. (Train-Test):  0.004157332 ; AIC: 48888.64 ; SBC:  49098.78
## Numero de variables:  30

## Modelo 6 modificado

## Train:  0.7228378 ; Test:  0.7192497 ; Dif. (Train-Test):  0.003588082 ; AIC: 49013.76 ; SBC:  49169.67
## Numero de variables:  22
```

Pese a aumentar tanto el AIC como el criterio SBC, la diferencia entre ambos R2 se ha visto reducido ligeramente. Por otro lado, si revisamos la nueva desviación típica y la comparamos con la del modelo original, apenas se ha visto aumentado (de 0.0131 a 0.0134):

```
## Modelo 6 original

##     modelo     media       sd
## 6 Modelo 6 0.7259522 0.013199

## Modelo 6 modificado

##                  modelo     media         sd
## 1 Modelo 6 (modificado) 0.7206228 0.01349691
```

Por otro lado, en relación con el modelo final nos encontramos con la interacción *CCAA:SameComAutonDiffProvPtge* cuyo p-valor sólo es significativo en las regiones de Andalucía y Navarra:

```
CCAAAN_NA:SameComAutonDiffProvPtge                0.998386   0.119905   8.326  < 2e-16 ***
CCAAAR_CM:SameComAutonDiffProvPtge                0.052157   0.096516   0.540   0.5889
CCAACAT_PV:SameComAutonDiffProvPtge               0.028968   0.082712   0.350   0.7262
CCAACV_EX_AS_BA_CA:SameComAutonDiffProvPtge       0.290994   0.130041   2.238   0.0253 *
CCAAMA_CA_RI_CE_ME_MU_GA:SameComAutonDiffProvPtge -0.214441   0.179183  -1.197   0.2314
```

En este caso, una posibilidad sería reagrupar la CCAA con menor representación (MA_CA_RI_CE_ME_MU_GA) con AR_CM (la más parecida en cuanto a media), pero pese a ello la calidad del modelo (tanto en AIC como SBC) empeora significativamente:

```
Recategorizando CCAA => AIC: 49072.51 ; SBC:  49208.09
Sin recategorizar    => AIC: 49013.76 ; SBC:  49169.67
```

Incluso podríamos eliminar directamente la interacción, de no ser por dos inconvenientes: en primer lugar, la interacción es muy significativa para una CCAA en concreto (AN_NA, con un p-valor muy pequeño), además de que eliminándolo obtendríamos un peor AIC y SBC, incluso mayor que en el caso anterior en el que sólo rectagorizamos la variable CCAA:

```
Eliminando la interaccion => AIC: 49089.39 ; SBC:  49211.41
Sin eliminar              => AIC: 49013.76 ; SBC:  49169.67
```

Una última posibilidad sería sustituir dicha interacción por algunas de las importantes obtenidas en el primer modelo de regresión. Sin embargo, muchas de las interacciones emplean variables muy correlacionadas con las del modelo. A modo de ejemplo: *CCAA:Age_0.4_Ptge* o *CCAA:CCAA:Age_under19_Ptge* están muy correlacionadas con el campo *Age_over65_pct*. Con respecto al resto de interacciones, no aportan ninguna mejoría. Por tanto, una vez eliminadas las variables correlacionadas, analizamos la importancia de los parámetros mediante la función *summary*:

```
Call:
lm(formula = as.formula(formula.final), data = data_train)

Residuals:
    Min      1Q  Median      3Q     Max 
-54.396  -6.140   0.114   6.481  43.811 

Coefficients:
                                          Estimate Std. Error t value Pr(>|t|)    
(Intercept)                              43.603867   2.291686  19.027  < 2e-16 ***
CCAAAN_NA                                19.700804   3.838238   5.133 2.94e-07 ***
CCAAAR_CM                                 4.882464   2.750402   1.775   0.0759 .  
CCAACAT_PV                               19.219723   3.550058   5.414 6.39e-08 ***
CCAACV_EX_AS_BA_CA                       13.785834   3.130045   4.404 1.08e-05 ***
CCAAMA_CA_RI_CE_ME_MU_GA                 16.909840   3.816258   4.431 9.53e-06 ***
Age_over65_pct                            0.138707   0.015271   9.083  < 2e-16 ***
SameComAutonPtge                          0.213710   0.026464   8.076 7.94e-16 ***
SameComAutonDiffProvPtge                 -0.136986   0.061436  -2.230   0.0258 *  
IndustryUnemploymentPtge                 -0.079685   0.013141  -6.064 1.40e-09 ***
ServicesUnemploymentPtge                 -0.039045   0.005643  -6.919 5.00e-12 ***
logxConstructionUnemploymentPtge         -0.254405   0.034783  -7.314 2.90e-13 ***
CCAAAN_NA:SameComAutonPtge               -0.529358   0.044973 -11.771  < 2e-16 ***
CCAAAR_CM:SameComAutonPtge               -0.162314   0.032771  -4.953 7.49e-07 ***
CCAACAT_PV:SameComAutonPtge              -0.813113   0.042966 -18.925  < 2e-16 ***
CCAACV_EX_AS_BA_CA:SameComAutonPtge      -0.344471   0.037092  -9.287  < 2e-16 ***
CCAAMA_CA_RI_CE_ME_MU_GA:SameComAutonPtge -0.257115   0.046650  -5.512 3.69e-08 ***
CCAAAN_NA:SameComAutonDiffProvPtge        0.998386   0.119905   8.326  < 2e-16 ***
CCAAAR_CM:SameComAutonDiffProvPtge        0.052157   0.096516   0.540   0.5889    
CCAACAT_PV:SameComAutonDiffProvPtge       0.028968   0.082712   0.350   0.7262    
CCAACV_EX_AS_BA_CA:SameComAutonDiffProvPtge  0.290994   0.130041   2.238   0.0253 *  
CCAAMA_CA_RI_CE_ME_MU_GA:SameComAutonDiffProvPtge -0.214441   0.179183  -1.197   0.2314    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.5 on 6474 degrees of freedom
Multiple R-squared:  0.7228,	Adjusted R-squared:  0.7219 
F-statistic:   804 on 21 and 6474 DF,  p-value: < 2.2e-16
```

Analizando la función *summary* salvo la interacción final, el resto de variables son prácticamente significativas. En relación con la última interacción, pese a que no todas las Comunidades Autónomas sean significativas, su importancia no deja de ser relevante. A modo de ejemplo, analicemos la salida obtenida en *modelEffectSizes*:

```
## lm(formula = as.formula(formula.final), data = data_train)
## 
## Coefficients
##                                  SSR df pEta-sqr dR-sqr
## (Intercept)               39949.4925  1   0.0530     NA
## CCAA                       6563.3025  5   0.0091 0.0025
## Age_over65_pct             9104.3681  1   0.0126 0.0035
## SameComAutonPtge           7196.5256  1   0.0100 0.0028
```

```
## SameComAutonDiffProvPtge           548.6345  1   0.0008 0.0002
## IndustryUnemploymentPtge          4057.7080  1   0.0056 0.0016
## ServicesUnemploymentPtge          5282.2642  1   0.0073 0.0020
## logxConstructionUnemploymentPtge  5903.2263  1   0.0082 0.0023
## CCAA:SameComAutonPtge            48723.8001  5   0.0638 0.0189
## CCAA:SameComAutonDiffProvPtge     9479.8504  5   0.0131 0.0037
##
## Sum of squared errors (SSE): 714404.4
## Sum of squared total  (SST): 2577567.7
```

Pese a que solo se pierda una pequeña proporción de R2 (0.0037), quisiera remarcar el el campo *pEta-sqr*, el cual indica **la varianza de la variable objetivo explicada por cada variable del modelo**. De hecho, **las Comunidades Autónomas, el porcentaje de población superior a 65 años, el porcentaje de población que reside en la misma CCAA junto con el porcentaje de población que reside en una provincia diferente son las que mayor cantidad de varianza explican**, con más de un 1 % en cada una de ellas (incluso en el caso de *CCAA:SameComAutonPtge* llegando a alcanzar más del 6 % de la varianza explicada). Por tanto, en el caso de *CCAA:SameComAutonDiffProvPtge* no podemos eliminarlo pese a la menor importancia de las variables, ya que de lo contrario perderíamos porcentaje de variabilidad explicada (cerca del 1.3 %).

Como conclusión final, tras elegir el modelo 6 como modelo ganador en el proceso de selección, así como una vez eliminadas aquellas variables con elevada correlación, obtenemos un modelo bastante significativo en prácticamente todos sus parámetros, en el que hemos podido comprobar a lo largo del proceso final de depuración que **el porcentaje de votos a la derecha se ve influido principalmente por la CCAA del municipio, el porcentaje de habitantes mayores a 65 años e incluso en función del porcentaje de personas que residan en la misma CCAA (sea en la misma provincia o no), factores que en muchos de los casos se han visto potenciados al emplear interacciones con las Comunidades Autónomas**. Por tanto, una vez evaluado, declaramos el modelo 6 depurado como modelo ganador:

```
ESTADISTICAS DEL MODELO FINAL:
Train:  0.7228378 ; Test:  0.7192497 ; Dif. (Train-Test):  0.003588082 ; AIC: 49013.76 ; SBC:  49169.67
Numero de variables:  22 ; sd: 0.01349691
```

2.4 Interpretación de los coeficientes de dos variables

Finalmente, interpretaremos los coeficientes de dos variables obtenidas en el modelo.

1. **CCAACAT_PV** (Cataluña y País Vasco): 19.22. Es decir, el porcentaje de votos a la derecha aumenta en un 19.22 % aproximadamente si la CCAA a la que pertenece el municipio es Cataluña o País Vasco con respecto a la Comunidad Autónoma de referencia (Castilla y León).
2. **logxConstructionUnemploymentPtge**: -0.25. Es decir, por cada incremento unitario en el porcentaje de desempleados en el sector de la construcción, el porcentaje de votos a la derecha se ve reducido en un 0.25 %. Por tanto, aquellos municipios con mayor porcentaje de paro en la construcción **afectarán negativamente al voto de la derecha**.

3. Construcción del modelo de regresión logística

Una vez construido el modelo de regresión lineal, continuamos con el modelo de regresión logística. En primer lugar, y al igual que en el apartado anterior, elaboramos un primer modelo con todas las variables e interacciones (aunque no se trate del modelo definitivo):

```r
formInt.bin<-formulaInteracciones(input_bin, 1)
modelo1.bin<-glm(formInt.bin,data=data_train.bin, family = binomial)
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
mostrar.estadisticas(modelo1.bin, data_train.bin, data_test.bin, "glm", "varObjBin")
```

```
## Train:  -10.44845 ; Test:  -11.01544 ; Dif. (Train-Test):  0.5669854 ; AIC: 99295.52 ; SBC:  101356.3
## Numero de variables:  304
```

Como podemos observar en la salida anterior, con un total de 304 parámetros el modelo no logra converger principalmente por un motivo: **existen demasiadas variables**, lo cual se traduce además en valores pseudo-R2 negativos, es decir, **la inclusión de demasiadas variables está penalizando la calidad del modelo**. Con respecto a las interacciones, debemos recoger únicamente aquellas con mayor relevancia. Por ello, ejecutamos la función *impVariablesLog*:

```
importancia.var <- impVariablesLog(modelo1.bin, "varObjBin", data_train.bin)
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -10.94997  -1.13104  -0.05853  -0.86781   1.29203   4.63291
```

Analizando la salida obtenida en el *summary* anterior, cabe destacar la mediana obtenida (-0.05), es decir, **un 50 % de las variables del modelo presentan una importancia negativa con respecto al modelo de regresión**, es decir, están penalizando las estimaciones obtenidas. Por el contrario, si nos fijamos en el tercer cuartil, un 25 % de las variables aportan la mayor importancia al modelo, **sobresalen** con respecto al resto. Por tanto, de cara a un segundo modelo (además de reducir el coste computacional en la selección clásica), **filtramos aquellas variables cuya importancia sea mayor al tercer cuartil (1.29)**:

```
variables.mas.imp <- importancia.var[which(importancia.var$V5 > 1.29203), "V2"]
# Ejemplo de algunas de las variables mas importantes (Top 5)
```

```
## [1] "IndustryUnemploymentPtge:ActividadPpal"
## [2] "Age_over65_pct:Densidad"
## [3] "CCAA:ForeignersPtge"
## [4] "CCAA:IndustryUnemploymentPtge"
## [5] "Densidad:prop_missings"
```

A primera vista, nos encontramos con que las variables más importantes corresponden con interacciones. A diferencia del modelo de regresión lineal, dichas interacciones no solo corrsponden con la Comunidad Autónoma, sino incluso con la Densidad o la Actividad Principal del municipio ¿Puede verse influido el voto a la Derecha por ambas variables categóricas en lugar de emplear únicamente la CCAA? Una vez recuperadas las interacciones más importantes, elaboramos un segundo modelo junto con las variables originales:

```
modelo1.2.bin<-glm(formInt.bin,data=data_train.bin, family = binomial)
```

```
## Train:  0.4442207 ; Test:  0.4306761 ; Dif. (Train-Test):  0.0135446 ; AIC: 4990.907 ; SBC:  5668.802
## Numero de variables:  100
```

Como podemos observar, no solo hemos conseguido reducir el número de parámetros (de 304 a 100), sino además que los valores obtenidos tanto del conjunto de datos *train* como *test* se corresponden con valores comúnes en el pseudoR2 (del orden de 0.4); además de unos criterios AIC y SBC mucho menores, reduciendo de 99.295 a 4.990 en el caso de AIC, por ejemplo.

3.1 Selección de variables clásica

No obstante, pese a que el modelo consigue converger, continúa teniendo demasiados parámetros. Por tanto, partiendo de este último modelo realizamos una selección clásica del mismo modo que en la regresión lineal: empleando los criterios AIC-both, SBC-both, AIC-forward, SBC-forward, AIC-backward y SBC-backward, devolviendo sus resultados en una tabla.

```
##              R.2.train  R.2.test    Diferencia      AIC      SBC N.Parametros
## AIC-both     0.4296977 0.4292625  0.0004352367 5008.097 5319.928           46
## SBC-both     0.4095392 0.4118443 -0.0023051484 5113.867 5195.214           12
## AIC-forward  0.4298391 0.4294172  0.0004218693 5008.879 5327.489           47
## SBC-forward  0.4095392 0.4118443 -0.0023051484 5113.867 5195.214           12
## AIC-backward 0.4425879 0.4287903  0.0137976379 4960.982 5489.739           78
## SBC-backward 0.4197849 0.4131002  0.0066846765 5043.548 5185.906           21
```

Analizando los resultados obtenidos en la selección clásica, los modelos 1 y 5 (AIC-both y AIC-backward) presentan un criterio AIC menor, aunque con un elevado número de parámetros, especialmente en el quinto modelo, donde incluso hay una mayor diferencia entre ambos pseudo-R2. Por el contrario, los modelos 2 y 6 (SBC-both y SBC-backward) ofrecen un menor número de variables, además de un valor SBC significativamente menor (5195 y 5185, respectivamente). Por tanto, pese al menor valor pseudo-R2 que presentan, los modelos 2 y 6 ofrecen unos resultados muy similares a los modelos 1 y 5, con una diferencia de tan solo 0.01 en el R2, empleando tan solo 12 y 21 parámetros, respectivamente. Por tanto, **todo apunta a los modelos 2 y 6 como posibles modelos candidatos**. Sin embargo, llama la atención la diferencia negativa entre los valores *train* y *test* del modelo 2. Por lo general, en cualquier modelo de aprendizaje automático el conjunto de datos de entrenamiento obtiene un mejor resultado en comparación con los datos de prueba. No obstante, puede ocurrir que los resultados en la validación/prueba sean ligeramente superiores a los datos de entrenamiento, en función del modo en el que se hayan dividido los datos (valor de la semilla). Dado que la partición ha sido aleatoria, puede ocurrir que el conjunto de entrenamiento sea más difícil de interpretar que los datos de validación, obteniendo resultados confusos. Por tanto, con los criterios AIC/SBC no resultan suficientes para decidir cual es el mejor modelo clásico, por lo que realizamos una **validación cruzada del mismo modo que en la regresión lineal**. De este forma podremos comprobar si los resultados son independientes en función de la partición de los datos: **si la desviación típica en los valores ROC es baja, significaría que la partición empleada en la selección clásica no sería la más adecuada. En caso contrario, podría tratarse de un posible sobreajuste en el modelo 2**:

```
##     modelo     media          sd
## 1 Modelo 1 0.8835252 0.008007286
## 2 Modelo 2 0.8787965 0.008992951
## 3 Modelo 3 0.8834541 0.007993940
## 4 Modelo 4 0.8787965 0.008992951
## 5 Modelo 5 0.8873403 0.007713627
## 6 Modelo 6 0.8825716 0.008435706
```

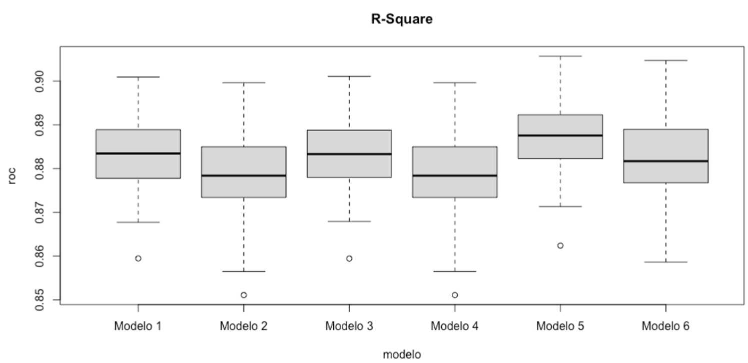


Figure 4: Validación cruzada en el modelo de selección aleatoria

Analizando tanto la tabla como los diagramas de caja, observamos que la desviación típica en el modelo 2 no es muy significativa, por lo que puede que la división empleada en la selección clásica no haya sido la más adecuada. Por lo general, tanto la media como las desviaciones típicas obtenidas son muy similares, con mejor resultado en los modelos 1, 3 y 5, además de una mejor bondad media en el valor ROC. No obstante, tanto el criterio SBC como el menor número de parámetros nos lleva a elegir los modelos 2 y 6. Como última comparación, analizamos el promedio de los p-valores obtenidos en cada modelo:

```
## Modelo  1
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00000 0.01102 0.09664 0.19209 0.26612 0.96024
## Modelo  2
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0000000 0.0000000 0.0571055 0.0000841 0.6845788
## Modelo  3
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00000 0.01869 0.11223 0.19670 0.27570 0.95923
## Modelo  5
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00000 0.01833 0.13422 0.23408 0.38621 0.97194
## Modelo  6
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.000e+00 0.000e+00 1.856e-05 2.128e-02 2.115e-02 1.551e-01
```

Pese a tener mejores medias y desviaciones típicas, **muchas de las variables en los modelos 1, 3 y 5 no son significativas**. A modo de ejemplo, la mediana en cada uno de ellos indica que un 50 % de los coeficientes no disminuye su p-valor de 0.05 (poca significancia), por lo que los descartamos. Como comparación final, el modelo 6 ha demostrado tener, en la validación cruzada, una media superior al modelo 2 (0.88 frente a 0.87). En relación a los p-valores, bien es cierto que la mayoría de las variables del segundo modelo son más relevantes (tercer cuartil = 8e-05 frente a 0.021 del modelo 6). No obstante, de cara a una comparación con los modelos aleatorios **emplearemos el**

modelo 6, ya que es posible que algunas de las variables sean poco significativas o estén correlacionadas entre sí, por lo que podremos eliminar dichos parámetros y decantarnos por un modelo definitivo.

3.2 Selección de variables aleatoria

Una vez realizada la selección clásica, elaboramos los modelos aleatorios a partir de la combinación de **todas las variables y sus interacciones**, un proceso computacionalmente más costoso pero que permite comprobar si hemos omitido alguna interacción relevante en los primeros pasos:

```
## MODELO 6 CLASICO

## Train:  0.4197849 ; Test:  0.4131002 ; Dif. (Train-Test):  0.006684677 ; AIC: 5043.548 ; SBC:  5185.906
## Numero de variables:  21

## MODELOS ALEATORIOS (TOP 3)

## Modelo aleatorio  1
## Train:  0.4092791 ; Test:  0.4070475 ; Dif. (Train-Test):  0.002231567 ; AIC: 5116.109 ; SBC:  5197.456
## Numero de variables:  12
## Modelo aleatorio  2
## Train:  0.4046166 ; Test:  0.4083817 ; Dif. (Train-Test):  -0.003765128 ; AIC: 5150.301 ; SBC:  5211.312
## Numero de variables:  9
## Modelo aleatorio  3
## Train:  0.4078011 ; Test:  0.4075987 ; Dif. (Train-Test):  0.000202435 ; AIC: 5126.85 ; SBC:  5201.418
## Numero de variables:  11
```

3.3 Selección y justificación del modelo ganador

Analizando el top 3 modelos aleatorios, llama la atención el primer modelo, con muchos menos parámetros que el modelo 6, aunque con valores AIC y SBC ligeramente superiores y un pseudo-R2 menor. No obstante, si analizamos las desviaciones típicas:

```
##     modelo     media          sd
## 1 Modelo 1 0.8825716 0.008435706
## 2 Modelo 2 0.8798386 0.008783255
## 3 Modelo 3 0.8760779 0.009312567
## 4 Modelo 4 0.8790228 0.008979965
```

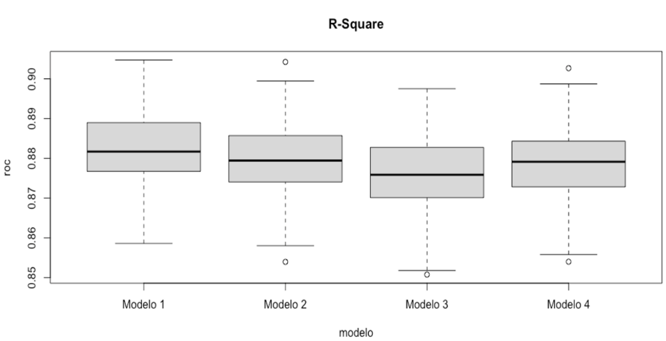


Figure 5: Validación cruzada modelo 6 + modelos de selección aleatoria

En ambos casos, tanto los valores medios de la curva ROC como la desviación típica son muy similares, con una mayor ventaja en el modelo 6 (modelo 1 en la imagen). No obstante, el menor número de parámetros aporta cierta ventaja al modelo 1 aleatorio (modelo 2 en la imagen). Como última prueba, dado que en ambos modelos se encuentran los campos *Age_over65_pct* y *PobChange_pct*, en función del grado de importancia eliminamos una u otra de ambos modelos, comparando de nuevo los resultados finales (en ambos casos eliminamos *PobChange_pct* dada su menor importancia):

```r
impVariablesLog(estadisticas.modelos.bin[6]$`SBC-backward`, "varObjBin", data_train.bin)
impVariablesLog(glm(paste0("varObjBin~",rownames(modelos.aleatorios)[1]), data_train.bin,
                    family = binomial), "varObjBin", data_train.bin)
```

Variables modelo 6	Importancia	Variables modelo 2	Importancia
AgricultureUnemploymentPtge	0.00137	PobChange_pct	0.00147
PobChange_pct	0.00150	AgricultureUnemploymentPtge	0.00226

```
               prop_missings 0.00216             Age_over65_pct 0.00360
              Age_over65_pct 0.00311                   Densidad 0.00537
  ForeignersPtge:ActividadPpal 0.00314            ForeignersPtge 0.00865
 CCAA:IndustryUnemploymentPtge 0.00658                       CCAA 0.33007
```

```
## Correlacion entre Age_over65_pct y PobChange_pct:  -0.5292537
```

```
## Modelo 6 clásico (modificado):
```

```
## Train:  0.4182844 ; Test:  0.4139388 ; Dif. (Train-Test):  0.004345615 ; AIC: 5054.482 ; SBC:  5190.061
## Numero de variables:  20
```

```
## Modelo 1 aleatorio (modificado):
```

```
## Train:  0.4078011 ; Test:  0.4075987 ; Dif. (Train-Test):  0.000202435 ; AIC: 5126.85 ; SBC:  5201.418
## Numero de variables:  11
```

En ambos casos, pese a eliminar las variables con mayor correlación, **en términos AIC/SBC además de un mayor pseudoR2** continúa dando una mayor ventaja al modelo 6 clásico. Por otro lado, si comparamos las nuevas medias y desviaciones típicas:

```
##                            modelo     media          sd
## 1          Modelo 6 (modificado) 0.8819148 0.008461733
## 2 Modelo 1 aleatorio (modificado) 0.8790228 0.008979965
```

En ambos casos, continuan favoreciendo al modelo 6. No obstante, el modelo 1 parece tratarse de un modelo mucho más sencillo, pues si nos fijamos en la salida anterior de *impVariablesLog*, **no emplea ninguna interacción**. De hecho, mientras que el modelo 6 interactúa el porcentaje de extranjeros con la Actividad principal, así como la CCAA con el porcentaje de desempleo en la industria, el modelo 1 utiliza el campo Densidad así como el porcentaje de desempleados en el sector agrícola. En general, tanto el campo CCAA sin interactuar como el nivel de Densidad en el municipio **suponen una mayor importancia en comparación con las interacciones del modelo 1**. Si analizamos nuevamente los p-valores:

```
## Modelo 6 clásico (modificado):
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0000000 0.0000337 0.0249546 0.0305279 0.1513579
```

```
## Modelo 1 aleatorio (modificado):
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0000000 0.0000000 0.0488137 0.0000047 0.5369334
```

Gran parte de las variables en el modelo 1 (concretamente el 75 %) aportan mucha más importancia que las variables del modelo 6: 4.7e-06 en comparación con 0.03. Por tanto, pese a que el modelo 6 aporte un mejor pseudoR2 (0.41 frente a 0.40), además de un mejor criterio AIC-SBC, los resultados del *summary* indican que la importancia de sus variables es menos significativa con respecto al modelo 1, un modelo que, pese a no utilizar ninguna interacción, obtiene unos valores pseudoR2 muy similares al modelo clásico (la diferencia entre ambos R2 es de apenas 0.01). Una posibilidad sería **comprobar si existen interacciones que mejoren las estadísticas del modelo 1**, concretamente las interacciones más importantes que pudimos extraer en el primer modelo:

	Interaccion	AIC	SBC	Dif. PseudoR2	sd
1	IndustryUnemploymentPtge:ActividadPpal	5119.72	5214.62	-0.00143	0.008905
2	Age_over65_pct:Densidad	5130.51	5218.64	-0.00019	0.008991
3	CCAA:ForeignersPtge	5121.14	5229.6	-0.00085	0.009038
4	CCAA:IndustryUnemploymentPtge	5083.37	5198.61	0.00715	0.008450
5	Densidad:prop_missings	5124.47	5219.38	-0.00203	0.008921
6	Sin interaccionar	5126.85	5201.41	0.00020	0.008979

Incluso añadiendo algunas de las interacciones más relevantes, en la mayoría de los casos se obtiene un valor pseudoR2 en el conjunto *test* ligeramente superior con respecto a los datos de entrenamiento (diferencia negativa). De hecho, aunque no tuviéramos en cuenta los valores R2, tanto el criterio AIC como la desviación típica no parecen reducirse significativamente al incluir una interacción, incluso en algunos casos puede llegar a aumentar con respecto a los valores originales. Por tanto, **las interacciones**, pese a la mayor importancia que presentaban en los primeros modelos, **no parecen aportar mejoría alguna**.

Como conclusión final, nos encontramos ante dos posibles modelos cuyo poder predictivo es similar (0.41-0.40), por lo que siguiendo el principio de parsimonia *en igualdad de condiciones, la explicación más sencilla suele ser la más probable*. Dado que el modelo 1 aleatorio ofrece un menor número de variables, **lo declaramos como modelo ganador**.

A continuación, analizamos los coeficientes del modelo ganador mediante la función *summary*:

```
Call:
glm(formula = formula.final.bin.aleatorio, family = binomial,
    data = data_train.bin)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7761  -0.1598   0.3932   0.6425   3.4267

Coefficients:
                            Estimate Std. Error z value Pr(>|z|)
(Intercept)                 1.679445   0.168185   9.986  < 2e-16 ***
Age_over65_pct              0.016838   0.003780   4.454 8.42e-06 ***
AgricultureUnemploymentPtge -0.016630  0.003613  -4.603 4.16e-06 ***
CCAAAN_NA                  -3.357578   0.127897 -26.252  < 2e-16 ***
CCAAAR_CM                  -1.572530   0.107108 -14.682  < 2e-16 ***
CCAACAT_PV                 -7.715219   0.398165 -19.377  < 2e-16 ***
CCAACV_EX_AS_BA_CA         -1.822037   0.118062 -15.433  < 2e-16 ***
CCAAMA_CA_RI_CE_ME_MU_GA   -0.104502   0.169245  -0.617    0.537
DensidadBaja                0.540913   0.118780   4.554 5.27e-06 ***
DensidadAlta                1.022174   0.175629   5.820 5.88e-09 ***
ForeignersPtge              0.050303   0.006158   8.169 3.11e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 8620.2  on 6495  degrees of freedom
Residual deviance: 5104.8  on 6485  degrees of freedom
AIC: 5126.8

Number of Fisher Scoring iterations: 7
```

Analizando los coeficientes, llama la atención las provincias de Madrid, Cantabria, Rioja, Ceuta, Melilla, Murcia y Galicia (MA__CA__RI__CE__ME__MU__GA), cuyo p-valor corresponde con el máximo de todo el modelo (0.537). Al tratarse de la categoría con menor número de variables (816), puede ocurrir que la categoría tenga una menor representación con respecto al resto de Comunidades.

```
ESTADISTICAS DEL MODELO FINAL:
Train:  0.4078011 ; Test:  0.4075987 ; Dif. (Train-Test):  0.000202435 ; AIC: 5126.85 ; SBC:  5201.418
Numero de variables:  11 ; sd: 0.008979965
```

3.4 Selección del punto de corte óptimo

Una vez elegido el modelo final, debemos **evaluar cual es el punto de corte que ofrece un mejor resultado**. Para ello, obtenemos tanto el punto de corte que maximice la tasa de aciertos como el índice de Youden, generando una rejilla con todos los posibles puntos de corte (de 0 a 1 en intervalos de 0.01):

```
rejilla$posiblesCortes[which.max(rejilla$Accuracy)] # Maximiza tasa aciertos
```

```
## [1] 0.54
```

```
rejilla$posiblesCortes[which.max(rejilla$Youden)] # Maximiza indice Youden
```

```
## [1] 0.59
```

Una vez obtenidos ambos puntos de corte, comparamos los porcentajes recuperados de la matriz de confusión:

```
sensEspCorte(modelo.final.bin.aleatorio,data_test.bin,"varObjBin",0.54,"1") # Max. tasa aciertos
```

```
##       Accuracy    Sensitivity    Specificity Pos Pred Value Neg Pred Value
##      0.8367221      0.9404762      0.6666667      0.8222029      0.8723404
```

```r
sensEspCorte(modelo.final.bin.aleatorio,data_test.bin,"varObjBin",0.59,"1") # Indice Youden
```

```
##       Accuracy    Sensitivity    Specificity Pos Pred Value Neg Pred Value
##      0.8305607      0.9057540      0.7073171      0.8353156      0.8207547
```

Analizando los porcentajes obtenidos, mediante el primer índice se consigue maximizar la sensitividad o tasa de verdaderos positivos, es decir, de cada 100 municipios en los que el modelo considera que hay un mayor número de votos a la derecha, 94 de ellos son verdaderos positivos. Por otro lado, no solo es capaz de maximizar la tasa de sensitividad, sino además el valor predictivo negativo: de cada 100 municipios en los que el modelo considera que no hay mayor número de votos a la derecha, ha predicho correctamente 87 de ellos (reduciendo el número de falsos negativos). Sin embargo, el objetivo del proyecto (tal y como se mencionó al comienzo de la memoria) no es solo conseguir que el modelo sea capaz de acertar en qué municipios resulta ganador la derecha, sino además **ser capaz de acertar también (en la medida de los posible) aquellos municipios en los que no**. A modo de ejemplo, el primer índice es capaz de acertar con alta precisión qué municipios votan a la derecha, mejorando tanto el porcentaje de verdaderos positivos (94 %) como además reducir el número de falsos negativos (VPN = 87 %). No obstante, la menor sensitividad que presenta (66 %) refleja que el modelo "pasa por alto" un elevado número de municipios que deberían considerarse 0 (no gana la derecha), pero que los está clasificando como municipios donde gana la derecha (1): **aumenta el número de falsos positivos**. Por el contrario, el índice de Youden, aunque con una sensitividad y un valor predictivo negativo menor, consigue reducir el número de falsos positivos, acertando en algo más del 70 % de los municipios en los que la derecha no resulta ganador. Por tanto, sacrificando la tasa de sensitividad y el aumento de falsos negativos, **escogemos el índice de Youden como punto de corte óptimo, pues permite no solo acertar en algo más del 90 % aquellos municpios en los que verdaderamente gana la derecha, sino además maximizar la tasa de verdaderos positivos, con cerca del 70 %**.

Tras elegir el punto de corte óptimo, observamos que las medidas de clasificación son muy similares en ambos conjuntos de datos (entrenamiento y prueba), con porcentajes ligeramente superiores en el entrenamiento.

```r
sensEspCorte(modelo.final.bin.aleatorio,data_train.bin,"varObjBin",0.59,"1")
```

```
##       Accuracy    Sensitivity    Specificity Pos Pred Value Neg Pred Value
##      0.8348214      0.9090458      0.7131247      0.8385917      0.8270500
```

```r
sensEspCorte(modelo.final.bin.aleatorio,data_test.bin,"varObjBin",0.59,"1")
```

```
##       Accuracy    Sensitivity    Specificity Pos Pred Value Neg Pred Value
##      0.8305607      0.9057540      0.7073171      0.8353156      0.8207547
```

En relación con la curva ROC, los valores tanto en el entrenamiento como prueba son muy similares, con porcentajes muy cercanos al 90 % en ambos casos:

```
## Train AUC:  0.8799913 ; Test AUC:  0.8769293
```

3.5 Interpretación de los coeficientes de dos variables

Por último, interpretamos los coeficientes de dos variables incluidas en el modelo:

1. **DensidadAlta**: 1.022. Es decir, **el ODD de que en un municipio, cuya densidad de población es alta (> 5 hab./ha), resulte ganador la derecha es** $e^{1.022} = 2.78$ **veces mayor que el ODD de un municipio cuya densidad sea Muy Baja (< 1 hab./ha)**.
2. **AgricultureUnemploymentPtge**: -0.016. Es decir, ___por cada unidad en la que se decrementa el porcentaje de desempleados en el sector agrario, la ODD de que el municipio resulte ganador la derecha aumenta en un $e^{-0.016} = 0.98$ %. Es decir, **el ODD de que en el municipio gane la derecha aumenta cuanto menos desempleados en el sector agrario haya**.