Master en Big Data y Data Science

Imagen que contiene Logotipo

Descripción generada automáticamente

**Trabajo Fin de Master**

**Predicción y descubrimiento de conocimiento en la base de datos census income**

Ricardo Alcañiz Frutos

*Curso académico 2019-2020*

Contenido

[1. Introducción y objetivos 3](#_Toc57469422)

[2. Base de datos 3](#_Toc57469423)

[2.1. Descripción de la base de datos 3](#_Toc57469424)

[2.2. Lectura, limpieza, incorporación de variables e imputación de valores faltantes 4](#_Toc57469425)

[2.2.1. Lectura y limpieza de datos 4](#_Toc57469426)

[2.2.2. Incorporación de ratio de desempleo por nivel educativo 5](#_Toc57469427)

[2.2.3. Valores faltantes 7](#_Toc57469428)

[3. Análisis descriptivo 10](#_Toc57469429)

[3.1. Exploración descriptiva de los datos 10](#_Toc57469430)

[3.1.1. Exploración descriptiva a nivel de factor 11](#_Toc57469431)

[3.1.2. Exploración descriptiva variable continuas 18](#_Toc57469432)

[4. Interacción entre los factores 21](#_Toc57469433)

[5. Selección de variables 22](#_Toc57469434)

[6. Fusión de las categorías no significativas 25](#_Toc57469435)

[6.1. Education 25](#_Toc57469436)

[6.2. Marital status 27](#_Toc57469437)

[6.3. Relationship 30](#_Toc57469438)

[6.4. Occupation 31](#_Toc57469439)

[6.5. workclass 34](#_Toc57469440)

[6.6. capital.gain 36](#_Toc57469441)

[6.7. capital.loss 37](#_Toc57469442)

[7. Discretización de variables y equilibrado de la muestra 37](#_Toc57469443)

[8. Entrenamiento de los algoritmos 41](#_Toc57469444)

[8.1. CART 43](#_Toc57469445)

[8.2. Redes neuronales 46](#_Toc57469446)

[8.3. Random Forest 49](#_Toc57469447)

[8.4. Vecimos más próximos 52](#_Toc57469448)

[8.5. C5.0 55](#_Toc57469449)

[8.6.1. Gradient Boosting 58](#_Toc57469450)

[8.6.2. Gradient Boosting 61](#_Toc57469451)

[9. Comparativa de modelos 64](#_Toc57469452)

[10. Elección del modelo e interpretación de los resultados 68](#_Toc57469453)

[11. Conclusiones finales 69](#_Toc57469454)

[12. Referencias bibliográficas 70](#_Toc57469455)

## 1. Introducción y objetivos

El presente trabajo se encuadra dentro del ámbito socioeconómico y toma como base de datos la referenciada en [1], la cual contiene 48842 registros del censo estadounidense con datos desagregados a nivel individual, incluyendo diversos aspecto de índole social y económica tal y como se detallan en el epígrafe 2.1, junto con una variable dependiente de tipo dicotómico indicando si se supera un cierto umbral de ingresos (50k). Así, el objetivo primordial es resolver el problema de clasificación que define la propia base de datos. Además, se explorará la estructura predictiva de los modelos cuya conclusión es definir unos indicadores sencillos, pero significativos en la búsqueda del perfil por encima del umbral de ingresos establecido en la base de datos. Lo interesante es que dichos indicadores pueden servir de entrada a otros estudios, particularmente al propio análisis de datos. En efecto, una vez establecidos las métricas que caracterizan el objeto de estudio, puede plantearse más claramente una nueva representación de la base de datos, reflejando de manera más precisa las métricas o patrones encontados. Estas segunda y sucesivas iteraciones no se llevarán a cabo. Por lo tanto, las preguntas a responder son:

- ¿Podemos conseguir, para la base de datos dada, un predictor con un nivel de precisión aceptable?  
- Si es posible resolver el primer problema, ¿podemos extraer unos indicadores claros que nos manifiesten la estructura predictiva y arrojen luz sobre los factores más importantes para la clasificación de las personas, según sus ingresos?

Sin embargo, ambos objetivos no tienen porqué ir necesariamente en la misma dirección. De hecho, resumir y agregar la información, objetivos deseables para obtener unas métricas lo más sencillas posible, puede tener consecuencias en la exactitud del modelo. Por tanto, dependiendo de la base de datos, puede incluso plantearse la opción de manejar diferentes representaciones de la base de datos para cada uno de los fines. Para este caso, mostramos cómo con una única representación se pueden satisfacer ambos objetivos simultáneamente.

## 2. Base de datos

### 2.1. Descripción de la base de datos

Definimos en la siguiente tabla la estructura de la base de datos:

##   
##   
## +----------------+------------+------------------------------+  
## | Variable | Tipo | Descripcion |  
## +================+============+==============================+  
## | Age | Continua | Edad de la persona |  
## +----------------+------------+------------------------------+  
## | Workclass | Categorica | Clase a la que pertenece |  
## +----------------+------------+------------------------------+  
## | fnlwgt | Continua | Peso asignado segun las |  
## | | | caracteristicas individuales |  
## +----------------+------------+------------------------------+  
## | education | Categorica | Categoria educativa |  
## +----------------+------------+------------------------------+  
## | education-num | Numerica | Ordinal asignado a cada |  
## | | | categoria educativa |  
## +----------------+------------+------------------------------+  
## | marital-status | Categorica | Estado civil |  
## +----------------+------------+------------------------------+  
## | occupation | Categorica | Ocupacion |  
## +----------------+------------+------------------------------+  
## | relationship | Categorica | Relacion |  
## +----------------+------------+------------------------------+  
## | race | Categorica | Raza |  
## +----------------+------------+------------------------------+  
## | sex | Categorica | Sexo |  
## +----------------+------------+------------------------------+  
## | capital-gain | Continua | Ganancia Capital |  
## +----------------+------------+------------------------------+  
## | capital-loss | Continua | Perdida Capital |  
## +----------------+------------+------------------------------+  
## | hours-per-week | Continua | Horas a la semana trabajadas |  
## +----------------+------------+------------------------------+  
## | native-country | Categorica | Pais de procedencia |  
## +----------------+------------+------------------------------+

## 2.2. Lectura, limpieza, incorporación de variables e imputación de valores faltantes

### 2.2.1. Lectura y limpieza de datos

La base de datos se da en sendos ficheros de entrenamiento y validación, por lo que procedemos a fusionarlos en un único dataset para realizar las transformaciones al conjunto de todos los datos. Por otro lado, éstos presentan unos pequeños problemas con los espacios en blanco, ya que se introducen de forma arbitraria y hace que se dupliquen categorías de los factores. Por ello, es necesario normalizarlas eliminando los espacios en blanco.

# Lectura de los datos de entrenamiento y validación y fusionado en un único dataset  
census.training <- read.table(file="adult.data", header=TRUE, sep=",")  
census.test <- read.table(file="adult.test", header=TRUE, sep=",")  
census <- union(census.training, census.test)  
  
# Normalización de las categorías y eliminación de espacios en blanco  
census$y <- gsub("\\.", "", trim(census$y))  
census$workclass <- trim(census$workclass)  
census$education <- trim(census$education)  
census$marital.status <- trim(census$marital.status)  
census$occupation <- trim(census$occupation)  
census$relationship <- trim(census$relationship)  
census$race <- trim(census$race)  
census$sex <- trim(census$sex)  
census$native.country <- trim(census$native.country)  
census$native.country <- as.factor(census$native.country)  
census$y <- as.factor(census$y)  
# Estrucutura del dataset tras normalizar  
str(census)

## 'data.frame': 48813 obs. of 15 variables:  
## $ age : int 39 50 38 53 28 37 49 52 31 42 ...  
## $ workclass : Factor w/ 9 levels "?","Federal-gov",..: 8 7 5 5 5 5 5 7 5 5 ...  
## $ fnlwgt : int 77516 83311 215646 234721 338409 284582 160187 209642 45781 159449 ...  
## $ education : Factor w/ 16 levels "10th","11th",..: 10 10 12 2 10 13 7 12 13 10 ...  
## $ education.num : int 13 13 9 7 13 14 5 9 14 13 ...  
## $ marital.status: Factor w/ 7 levels "Divorced","Married-AF-spouse",..: 5 3 1 3 3 3 4 3 5 3 ...  
## $ occupation : Factor w/ 15 levels "?","Adm-clerical",..: 2 5 7 7 11 5 9 5 11 5 ...  
## $ relationship : Factor w/ 6 levels "Husband","Not-in-family",..: 2 1 2 1 6 6 2 1 2 1 ...  
## $ race : Factor w/ 5 levels "Amer-Indian-Eskimo",..: 5 5 5 3 3 5 3 5 5 5 ...  
## $ sex : Factor w/ 2 levels "Female","Male": 2 2 2 2 1 1 1 2 1 2 ...  
## $ capital.gain : int 2174 0 0 0 0 0 0 0 14084 5178 ...  
## $ capital.loss : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ hours.per.week: int 40 13 40 40 40 40 16 45 50 40 ...  
## $ native.country: Factor w/ 42 levels "?","Cambodia",..: 40 40 40 40 6 40 24 40 40 40 ...  
## $ y : Factor w/ 2 levels "<=50K",">50K": 1 1 1 1 1 1 1 2 2 2 ...

### 2.2.2. Incorporación de ratio de desempleo por nivel educativo

La ley de de oferta y demanda es el principio que, con caracter general, marca el precio de los productos en el mercado. Este principio tiene su extensión al ámbito laboral. Así, a priori es de esperar que profesiones con un alto nivel de demanda y poca oferta tengan un salario alto, mientras que profesiones con más oferta que demanda ofrezcan salarios más modestos. En vista de este razonamiento, es interesante la incorporación de datos del nivel de paro sobre diferentes estratos para la serie temporal anual de 1994 en US A. De manera ideal, la incorporación a buscar es al mismo nivel de granularidad que los datos de nuestro dataset, esto es, tener los datos de la serie temporal agregados por las variables que componen la base de datos (clase trabajadora, raza, edad, etc). Éstos se han recogido de la oficina de estadísticas de EEUU, referenciada en [2]. Sin embargo, la incorporación de los datos para el año requerido tiene serias deficiencias ya que, en general, o están demasiado agregados, tienen demasiados valores faltantes o los datos maestros no son comparables con los de la base de datos. Por todo ello, sólo ha sido posible introducir la variable afectando a los distintos niveles educativos. Para ello, se han incorporado a un fichero los datos por las distintas categorías educativas y se ha creado una variable auxiliar en la base de datos para poder cruzar los datos usando dicha variable. Es evidente que esta forma de incorporar los datos no recoge toda la variabilidad posible del nivel del paro y puede introducir errores, por posible correlación con la variable educación. Sin embargo, como veremos, es una variable significativa y a tener en cuenta.

# Lectura del excel con los datos de paro anuales por estratos educativos  
EmploymentRateByEducationLevel <- read\_excel("UnemploymentRateByEducationLevel.xlsx")  
EmploymentRateByEducationLevel

## # A tibble: 4 x 2  
## CategoryEducationlevel UnploymentRatio  
## <chr> <dbl>  
## 1 LessThanHighSchoolDiploma 9.8  
## 2 HighSchoolGraduatesNoCollege 5.4  
## 3 SomeCollegeOrAssociateDegree 4.5  
## 4 BachelorDegreeAndHigher 2.6

# Incorporación de la categoría educativa para el cruce con los datos de la serie temporal  
census <- census %>% mutate(CategoryEducationlevel = case\_when(education == 'Preschool' ~ 'HighSchoolGraduatesNoCollege',  
 education == '1st-4th' ~ 'HighSchoolGraduatesNoCollege',  
 education == '5th-6th' ~ 'HighSchoolGraduatesNoCollege',  
 education == '7th-8th' ~ 'HighSchoolGraduatesNoCollege',  
 education == '9th' ~ 'HighSchoolGraduatesNoCollege',  
 education == '10th' ~ 'HighSchoolGraduatesNoCollege',  
 education == '11th' ~ 'HighSchoolGraduatesNoCollege',  
 education == '12th' ~ 'HighSchoolGraduatesNoCollege',  
 education == 'HS-grad' ~ 'HighSchoolGraduatesNoCollege',  
 education == 'Some-college' ~ 'SomeCollegeOrAssociateDegree',  
 education == 'Assoc-voc' ~ 'SomeCollegeOrAssociateDegree',  
 education == 'Assoc-acdm' ~ 'SomeCollegeOrAssociateDegree',  
 education == 'Bachelors' ~ 'BachelorDegreeAndHigher',  
 education == 'Prof-school' ~ 'BachelorDegreeAndHigher',  
 education == 'Masters' ~ 'BachelorDegreeAndHigher',  
 education == 'Doctorate' ~ 'BachelorDegreeAndHigher')  
 )  
  
# Cruce de los datos e incorporación del nivel de desempleo por categoría social  
census <- merge(census, EmploymentRateByEducationLevel, by="CategoryEducationlevel", all.x = TRUE)  
# Eliminamos la variable auxiliar  
census$CategoryEducationlevel <- NULL

### 2.2.3. Valores faltantes

#### 2.2.3.1. Estudio de los valores faltantes

Aunque algunos algoritmos, especialmente los basados en estructuras de árbol, tienen incorporados imputación automática de valores faltantes y puede ser interesante mantener dos versiones de datos de entrenamiento, esto es, una versión con los valores faltantes miputados y otra que no, siguiendo el planteamiento inical, sólo vamos a mantener una versión de la base de datos de entrenamiento y, por lo tanto, es necesario el tratamiento de los valores faltantes. El siguiente chunck realiza una representación de las variables con valores faltantes y su porcentaje:

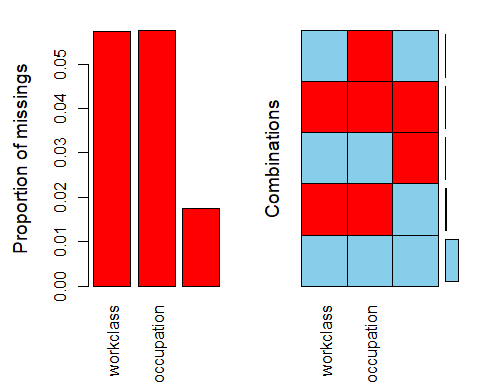
#Sustitución de valores faltantes por NA  
for(col in colnames(census)) {  
 census[, c(col)][census[, c(col)] == '?'] <- NA   
}  
# Número de líneas incompletas, es decir, con al menos una columna a null  
length(which(!complete.cases(census)))

## [1] 3619

# Columnas con al menos un valor nulo  
colWithNullValue <- census[, colnames(census) %in% colnames(census)[colSums(is.na(census)) > 0]]  
colnames(colWithNullValue)

## [1] "workclass" "occupation" "native.country"

# Porcentaje de valores nulos por columna  
aggr(colWithNullValue)



Como se ve de manera clara, las variables tienen un bajo índice de valores faltantes, un 5%, por lo que se está dentro de un umbral razonable para imputar los valores sin modificar demasiado las variables predictoras.

#### 2.2.3.2. Imputación de valores faltantes

De hecho, de entre todos los algoritmos de imputación de valores faltantes, hemos tomado uno que no alterase demasiado la distribución de los datos. En particular, hemos usado la moda para el país de procedencia, bajo la observación de que la mayoría de los datos son, como era de esperar en este caso, de origen estadounidense:

# Top n de ratio de nacionalidad  
census %>% filter(!is.na(native.country)) %>% group\_by(native.country) %>% summarise(n = n()) %>% mutate(freq = paste0(round(100 \* n/sum(n), 2), "%")) %>% arrange(desc(freq)) %>% top\_n(n = 5)

## Selecting by freq

## # A tibble: 6 x 3  
## native.country n freq   
## <fct> <int> <chr>   
## 1 United-States 43810 91.35%  
## 2 Mexico 947 1.97%   
## 3 Philippines 295 0.62%   
## 4 Germany 206 0.43%   
## 5 Canada 182 0.38%   
## 6 Puerto-Rico 184 0.38%

Por tanto, en este caso la moda tiene un alto valor a priori, por lo que usamos la mimsma para completar los valores faltantes, verificando que no se alteran significativamente los datos:

# Completamos los valores nulos usando la moda  
native.country.mode <- census %>% filter(!is.na(native.country)) %>% group\_by(native.country) %>% summarise(n = n()) %>% mutate(freq = paste0(round(100 \* n/sum(n), 2), "%")) %>% arrange(desc(freq)) %>% top\_n(n = 1) %>% dplyr::select(native.country)

## Selecting by freq

# imputación de la moda a los valores faltantes   
census[, c('native.country')][is.na(census[, c('native.country')])] <- as.factor(as.matrix(native.country.mode))  
  
  
# Top n de ratio de nacionalidad  
census %>% group\_by(native.country) %>% summarise(n = n()) %>% mutate(freq = paste0(round(100 \* n/sum(n), 2), "%")) %>% arrange(desc(freq)) %>% top\_n(n = 5)

## Selecting by freq

## # A tibble: 5 x 3  
## native.country n freq   
## <fct> <int> <chr>  
## 1 United-States 44666 91.5%  
## 2 Mexico 947 1.94%  
## 3 Philippines 295 0.6%   
## 4 Germany 206 0.42%  
## 5 Puerto-Rico 184 0.38%

Usando el mismo principio de mínima alteración en los datos, se ha seleccionado el algoritmo CART para la imputación de los restantes variables con valores nulos:

# Top n de ratio de clase trabajadora   
census %>% filter(!is.na(workclass)) %>% group\_by(workclass) %>% summarise(n = n()) %>% mutate(freq = paste0(round(100 \* n/sum(n), 2), "%")) %>% arrange(desc(freq))

## # A tibble: 8 x 3  
## workclass n freq   
## <fct> <int> <chr>   
## 1 Self-emp-not-inc 3861 8.39%   
## 2 Private 33879 73.63%  
## 3 Local-gov 3136 6.82%   
## 4 State-gov 1981 4.31%   
## 5 Self-emp-inc 1694 3.68%   
## 6 Federal-gov 1432 3.11%   
## 7 Without-pay 21 0.05%   
## 8 Never-worked 10 0.02%

# Top n de ratio de ocupacion  
census %>% filter(!is.na(occupation)) %>% group\_by(occupation) %>% summarise(n = n()) %>% mutate(freq = paste0(round(100 \* n/sum(n), 2), "%")) %>% arrange(desc(freq))

## # A tibble: 14 x 3  
## occupation n freq   
## <fct> <int> <chr>   
## 1 Machine-op-inspct 3019 6.56%   
## 2 Transport-moving 2355 5.12%   
## 3 Handlers-cleaners 2071 4.5%   
## 4 Farming-fishing 1487 3.23%   
## 5 Tech-support 1445 3.14%   
## 6 Protective-serv 983 2.14%   
## 7 Prof-specialty 6167 13.41%  
## 8 Craft-repair 6107 13.27%  
## 9 Exec-managerial 6084 13.22%  
## 10 Adm-clerical 5608 12.19%  
## 11 Sales 5504 11.96%  
## 12 Other-service 4919 10.69%  
## 13 Priv-house-serv 240 0.52%   
## 14 Armed-Forces 15 0.03%

# Método elegido para la imputación de variables faltantes  
#tmp.workclass.mice <- mice(census, m=5, maxit=5,meth='cart',seed='500')  
#saveRDS(tmp.workclass.mice, file = "tmp.workclass.mice.rds")  
tmp.workclass.mice <- readRDS(file = "tmp.workclass.mice.rds")  
  
census.tmp <- complete(tmp.workclass.mice, 1)  
  
# Top n de ratio de nacionalidad  
census.tmp %>% group\_by(workclass) %>% summarise(n = n()) %>% mutate(freq = paste0(round(100 \* n/sum(n), 2), "%")) %>% arrange(desc(freq))

## # A tibble: 8 x 3  
## workclass n freq   
## <fct> <int> <chr>   
## 1 Private 36678 75.14%  
## 2 Self-emp-not-inc 3861 7.91%   
## 3 Local-gov 3136 6.42%   
## 4 State-gov 1981 4.06%   
## 5 Self-emp-inc 1694 3.47%   
## 6 Federal-gov 1432 2.93%   
## 7 Without-pay 21 0.04%   
## 8 Never-worked 10 0.02%

# Top n de ratio de clase trabajadora  
census.tmp %>% group\_by(occupation) %>% summarise(n = n()) %>% mutate(freq = paste0(round(100 \* n/sum(n), 2), "%")) %>% arrange(desc(freq))

## # A tibble: 14 x 3  
## occupation n freq   
## <fct> <int> <chr>   
## 1 Machine-op-inspct 3216 6.59%   
## 2 Transport-moving 2481 5.08%   
## 3 Handlers-cleaners 2243 4.6%   
## 4 Farming-fishing 1559 3.19%   
## 5 Tech-support 1523 3.12%   
## 6 Protective-serv 1027 2.1%   
## 7 Prof-specialty 6414 13.14%  
## 8 Craft-repair 6378 13.07%  
## 9 Exec-managerial 6322 12.95%  
## 10 Adm-clerical 5997 12.29%  
## 11 Sales 5898 12.08%  
## 12 Other-service 5471 11.21%  
## 13 Priv-house-serv 269 0.55%   
## 14 Armed-Forces 15 0.03%

# Sustituímos el dataset temporal con los datos imputados por el data set real  
census <- census.tmp

# 3. Análisis descriptivo

## 3.1. Exploración descriptiva de los datos

Para tener una primera intuición de las relaciones entre las variables, se representan mediante resúmenes de datos agregados y gráficos el impacto que tiene las distintas variables en la separabilidad de las categorías de la variable respuesta. Se muestra la métrica normalizada número de casos con unos ingresos mayores a 50k respecto al total:

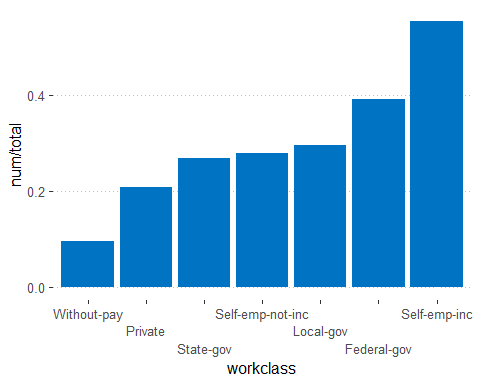
plot\_by\_factor<- function(dataset, factor, respuesta, clase\_ref) {  
  
census.dt <- as.data.table(dataset)  
cond <- quote(y == clase\_ref)  
# Cálculo por factor  
tmp <- merge(census.dt[, . (cnt = .N), by = c(factor, respuesta)][eval(cond), ],  
census.dt[, . (cnt = .N), by = c(factor)], by=factor, all.data=TRUE)[, c(factor, respuesta, "cnt.x", "cnt.y"), with=FALSE]  
  
names(tmp) <- c(factor, respuesta, "num", "total")  
  
  
print(tmp)  
  
# Plot por frecuencia  
h1 <- ggplot(as.data.frame(tmp), aes\_string(x = paste("reorder(", factor, ",", "num / total)", sep=" "), y = "num / total")) +  
 geom\_bar(fill = "#0073C2FF", stat = "identity") +  
 scale\_x\_discrete(name=factor, guide = guide\_axis(n.dodge=3)) +  
 theme\_pubclean()  
  
print(h1)  
}

### 3.1.1. Exploración descriptiva a nivel de factor

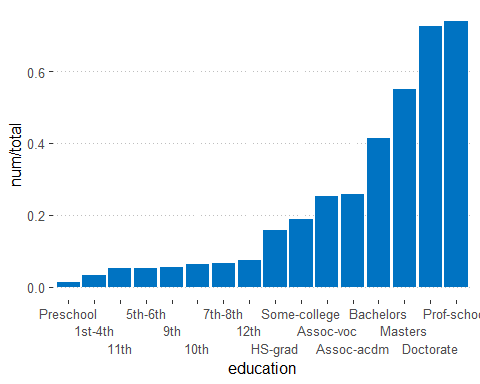
Se representan en primer lugar las variables categóricas:

factor.names <- names(Filter(is.factor, census))  
  
for(factor in factor.names[!factor.names %in% c ("y")]) {  
 plot\_by\_factor(census, factor, "y", ">50K")   
}

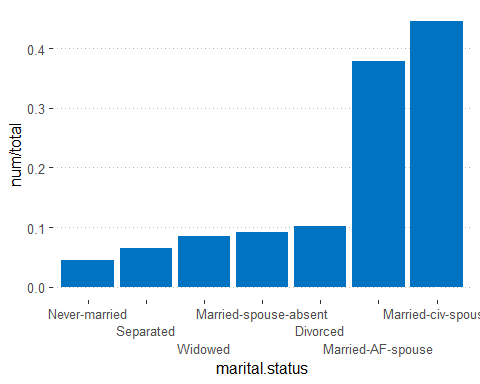
## workclass y num total  
## 1: Federal-gov >50K 561 1432  
## 2: Local-gov >50K 927 3136  
## 3: Private >50K 7650 36678  
## 4: Self-emp-inc >50K 938 1694  
## 5: Self-emp-not-inc >50K 1077 3861  
## 6: State-gov >50K 530 1981  
## 7: Without-pay >50K 2 21



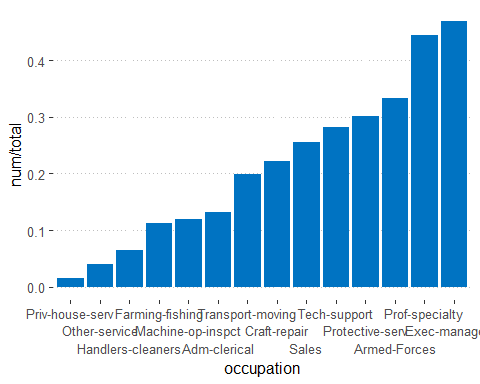
## education y num total  
## 1: 10th >50K 87 1389  
## 2: 11th >50K 92 1812  
## 3: 12th >50K 48 656  
## 4: 1st-4th >50K 8 245  
## 5: 5th-6th >50K 27 508  
## 6: 7th-8th >50K 62 954  
## 7: 9th >50K 41 756  
## 8: Assoc-acdm >50K 413 1601  
## 9: Assoc-voc >50K 522 2060  
## 10: Bachelors >50K 3313 8020  
## 11: Doctorate >50K 431 594  
## 12: HS-grad >50K 2502 15777  
## 13: Masters >50K 1459 2656  
## 14: Preschool >50K 1 82  
## 15: Prof-school >50K 617 834  
## 16: Some-college >50K 2062 10869



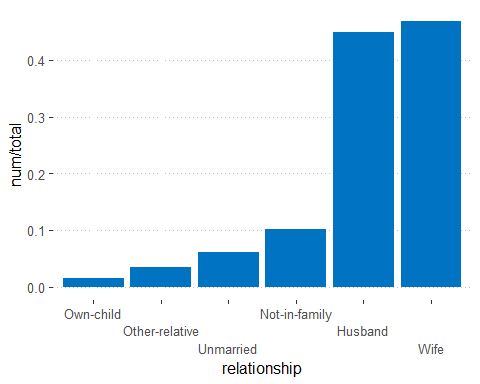
## marital.status y num total  
## 1: Divorced >50K 671 6630  
## 2: Married-AF-spouse >50K 14 37  
## 3: Married-civ-spouse >50K 9982 22372  
## 4: Married-spouse-absent >50K 58 628  
## 5: Never-married >50K 733 16098  
## 6: Separated >50K 99 1530  
## 7: Widowed >50K 128 1518



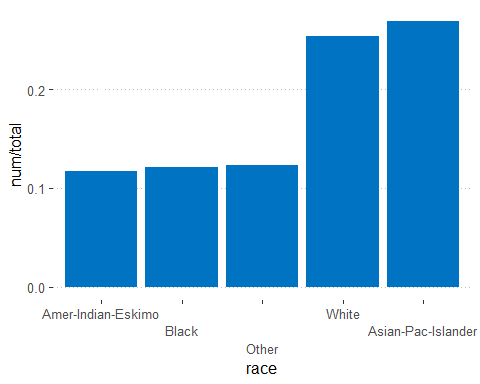
## occupation y num total  
## 1: Adm-clerical >50K 789 5997  
## 2: Armed-Forces >50K 5 15  
## 3: Craft-repair >50K 1414 6378  
## 4: Exec-managerial >50K 2967 6322  
## 5: Farming-fishing >50K 175 1559  
## 6: Handlers-cleaners >50K 145 2243  
## 7: Machine-op-inspct >50K 384 3216  
## 8: Other-service >50K 216 5471  
## 9: Priv-house-serv >50K 4 269  
## 10: Prof-specialty >50K 2845 6414  
## 11: Protective-serv >50K 310 1027  
## 12: Sales >50K 1508 5898  
## 13: Tech-support >50K 430 1523  
## 14: Transport-moving >50K 493 2481



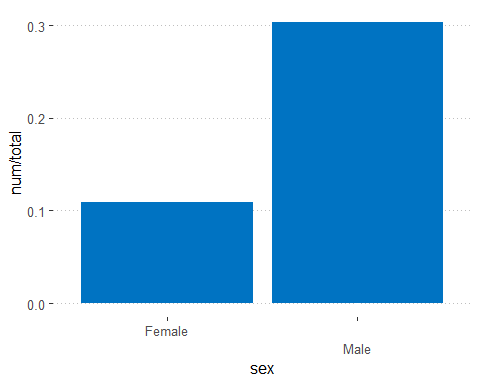
## relationship y num total  
## 1: Husband >50K 8844 19709  
## 2: Not-in-family >50K 1276 12567  
## 3: Other-relative >50K 52 1506  
## 4: Own-child >50K 111 7576  
## 5: Unmarried >50K 309 5124  
## 6: Wife >50K 1093 2331



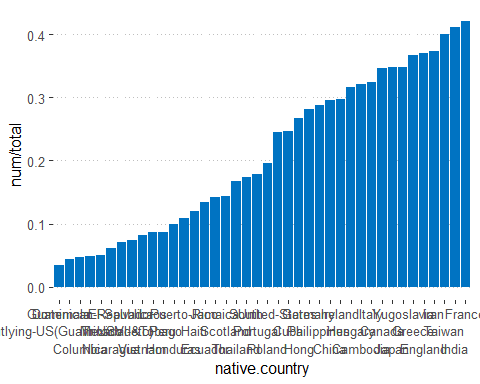
## race y num total  
## 1: Amer-Indian-Eskimo >50K 55 470  
## 2: Asian-Pac-Islander >50K 409 1518  
## 3: Black >50K 566 4683  
## 4: Other >50K 50 406  
## 5: White >50K 10605 41736



## sex y num total  
## 1: Female >50K 1769 16182  
## 2: Male >50K 9916 32631



## native.country y num total  
## 1: Cambodia >50K 9 28  
## 2: Canada >50K 63 182  
## 3: China >50K 36 122  
## 4: Columbia >50K 4 85  
## 5: Cuba >50K 34 138  
## 6: Dominican-Republic >50K 5 103  
## 7: Ecuador >50K 6 45  
## 8: El-Salvador >50K 11 155  
## 9: England >50K 47 127  
## 10: France >50K 16 38  
## 11: Germany >50K 58 206  
## 12: Greece >50K 18 49  
## 13: Guatemala >50K 3 86  
## 14: Haiti >50K 9 75  
## 15: Honduras >50K 2 20  
## 16: Hong >50K 8 30  
## 17: Hungary >50K 6 19  
## 18: India >50K 62 151  
## 19: Iran >50K 22 59  
## 20: Ireland >50K 11 37  
## 21: Italy >50K 34 105  
## 22: Jamaica >50K 15 106  
## 23: Japan >50K 32 92  
## 24: Laos >50K 2 23  
## 25: Mexico >50K 47 947  
## 26: Nicaragua >50K 3 49  
## 27: Outlying-US(Guam-USVI-etc) >50K 1 23  
## 28: Peru >50K 4 46  
## 29: Philippines >50K 85 295  
## 30: Poland >50K 17 87  
## 31: Portugal >50K 12 67  
## 32: Puerto-Rico >50K 20 184  
## 33: Scotland >50K 3 21  
## 34: South >50K 20 115  
## 35: Taiwan >50K 26 65  
## 36: Thailand >50K 5 30  
## 37: Trinadad&Tobago >50K 2 27  
## 38: United-States >50K 10912 44666  
## 39: Vietnam >50K 7 86  
## 40: Yugoslavia >50K 8 23  
## native.country y num total



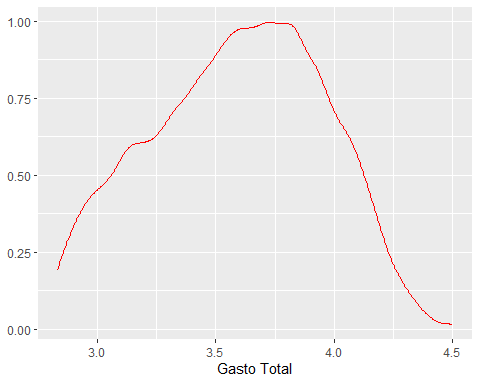
Se observa que:

* Existen categorías que tienen un número de casos muy pequeño y, por lo tanto, no resultan estadísticamente significativos, por lo que puede ser interesante fusionarlos con categorías con un número mayor de casos
* A priori, parece que el estado civil y la relación son las variables que mejor discrminan los la varviable respuesta

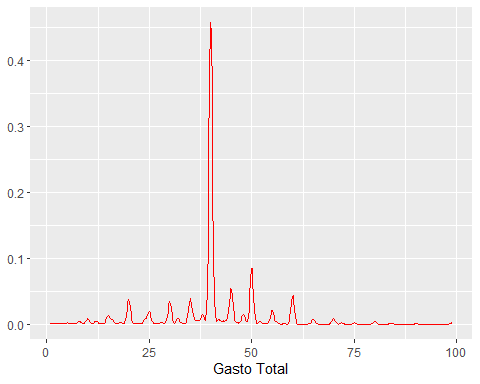
### 3.1.2. Exploración descriptiva variable continuas

Representamos la distribucón de cada una de las variables continúas:

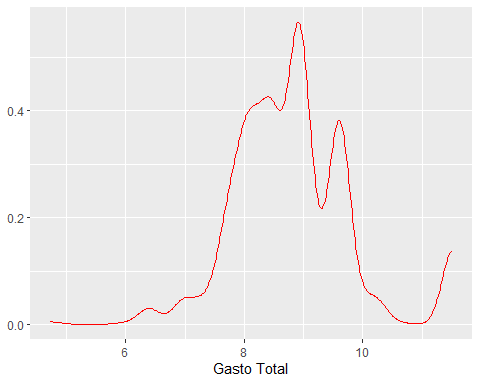
# age  
g2 <- ggplot(data = census,  
 aes(x = log(age))) +  
 geom\_line(stat="density", color = "red") +  
 xlab("Gasto Total") +  
 ylab(label = NULL)  
print(g2)



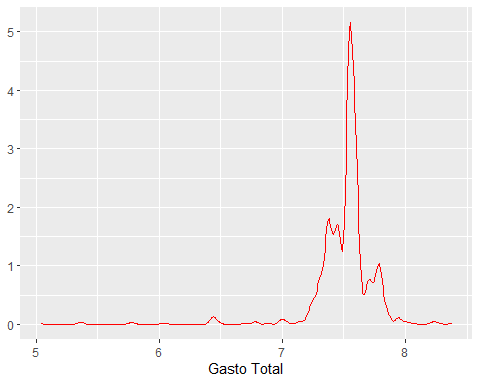
# hours.per.week  
g2 <- ggplot(data = census,  
 aes(x = hours.per.week)) +  
 geom\_line(stat="density", color = "red") +  
 xlab("Gasto Total") +  
 ylab(label = NULL)  
print(g2)



# capital.gain  
g2 <- ggplot(data = census[census$capital.gain > 0,],  
 aes(x = log(capital.gain))) +  
 geom\_line(stat="density", color = "red") +  
 xlab("Gasto Total") +  
 ylab(label = NULL)  
print(g2)



# capital.loss  
g2 <- ggplot(data = census[census$capital.loss > 0,],  
 aes(x = log(capital.loss))) +  
 geom\_line(stat="density", color = "red") +  
 xlab("Gasto Total") +  
 ylab(label = NULL)  
print(g2)



Como puede observarse, existen muchos factores que presentan pocos casos y que, por lo tanto, no son poco significativos desde el punto de vista estadístico. Para obtener mejores estimadores, vamos a estudiar la clusterización de los niveles de cada factor. Entre otras estrategias, se pueden plantear las siguientes:

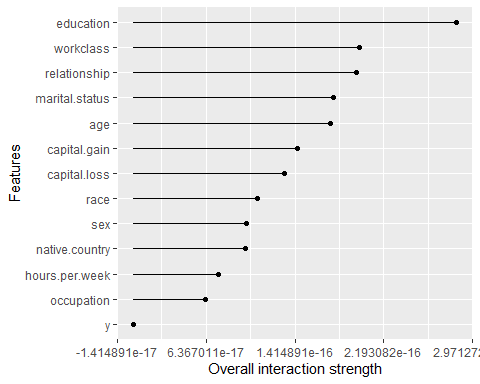
* Dada una variable endógena y otra exógena podemos dividir la varianza de la primera en la varianza intragrupos y varianza intergrupos. Minimizando la primera (esencialmente, maximizar el estadístico ), nos da un criterio para obtener grupos homogéneos de variables.
* Dado que trabajamos con variables nominales, podemos hacer uso del estadístico para obtener cluster de factores dependientes. Dada la tabla de contingencia construída a partir de la variable independiente y una variable exógena nominal, podemos ir buscando pares de categorías con un valor de p menor que un cierto umbral e ir fusionando éstas. Este proceso lo podemos continuar hasta obtener la definición de los cluster. Esto, tal y como se describre en [3], forma parte del algoritmo CHAID, por lo que haremos uso del mismo para la definición de los clústers de categorías.

Por otro lado, las variables de ingresos y gasto presentan unos máximos llamativos, que pueden sugerirnos la discretización de las variables en factores. Para la obtención de estos puntos de corte, usaremos el algoritmo CART, tal y como se explica en [4].

# 4. Interacción entre los factores

Antes de entrar en la transformación de los datos, se estudia si existe variabilidad explicada por las interacciones, tal y como se explica en [5]:

#glm.model <- glm(y ~ ., family="binomial", data = census)  
#mod <- Predictor$new(glm.model, data = census, y = census$y, class = "classification")  
#ia <- Interaction$new(mod)  
#saveRDS(ia, file = "ia.rds")  
ia <- readRDS(file = "ia.rds")  
ia$plot()



Como se observa, la variabilidad explicada por la interacción entre las variables es despreciable e igualmente, aunque no mostraremos el gráfico, las correlaciones entre las varibales no son lo suficientemente significativas como para intentar abordar métodos estadísticos de reumen de la información, como el análisis factorial. La variabilidad explicada es la aportada por cada una de las variables, sin tener en cuenta efectos de ningún tipo.

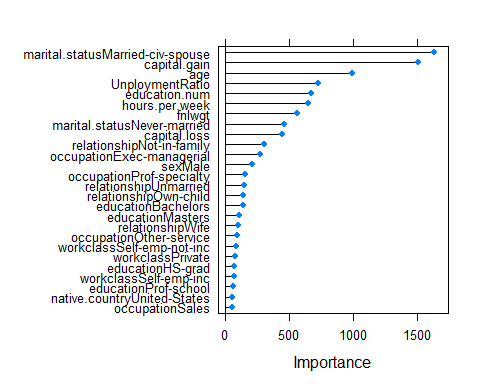
# 5. Selección de variables

Dada un estimador de una variable alatoria , sabemos que el error en la estimación puede descomponerse, en general, en una parte correspondiente al sesgo y otra correspondiente a la varianza del estimador. Es por ello que buscar un equilibrio entre ambos errores es de suma importancia y, en este sentido, la elección de las vraiables es clave. La omisión de variables relevantes afecta negativamente en el sesgo, mientras que la inclusión de variables redundantes o poco significativas aumentan la varianza del estimador. De las múltiples herramientas usadas para la selección de variables, usaremos el modelo random forest para la selección de varibles, conservando el 90% de la importancia de las variables seleccionadas.

# Transformación de la variable respuesta  
census$y <- as.factor(ifelse(census$y == ">50K", "yes", "no"))  
  
  
  
#glm.model <- glm(y ~ ., family="binomial", data = census)  
#summary(glm.model)  
  
#set.seed(999)  
#rfGrid <- expand.grid(mtry = 5:10)  
#clusterCPU <- registerDoParallel(cores=detectCores() - 1)  
#rf.fit <- train(y ~ .,   
# data = census,   
# method = "parRF",  
# preProc = c("center", "scale"),   
# trControl = train.control,  
# metric = "ROC",  
# tuneGrid = rfGrid  
# )  
#stopImplicitCluster()  
#rf.fit  
#saveRDS(rf.fit, file = "rf\_sv.rds")  
  
# Lectura del modelo entrenado  
rf\_sv.fit <- readRDS(file = "rf\_sv.rds")  
# Ranking de importancia de variables para RF  
vImp <- varImp(rf\_sv.fit, scale = FALSE)  
# Creación dataframe  
ImpMeasure <- as.data.frame(vImp$importance)  
ImpMeasure$Vars <- row.names(ImpMeasure)  
# Creaación de una clave surrogada, por orden de importancia de las variables  
ImpMeasure <- mutate(ImpMeasure[order(ImpMeasure$Overall, decreasing = TRUE), ], id=row\_number())  
# Cálculo de la importancia acumulada, mediante cross join  
CJ\_ImpMeasure <- tidyr::crossing(ImpMeasure, ImpMeasure)  
# Cálculo de la importancia absoluta (para cada fila i, sumar la importancia de todas las filas con un id menor o igual que i)  
CJ\_ImpMeasure <- CJ\_ImpMeasure[CJ\_ImpMeasure$id1 <= CJ\_ImpMeasure$id, ]  
CJ\_ImpMeasure <- CJ\_ImpMeasure %>% group\_by(id) %>% summarise(AccImportance = sum(Overall1))  
VarImportance <- merge(ImpMeasure, CJ\_ImpMeasure, by="id", all.x = TRUE)  
# Normalización  
VarImportance$AccPercentage <- VarImportance$AccImportance / sum(VarImportance$Overall)  
# Selección de las variables que conserven el 90% de la importancia de las variables  
VarImportance[VarImportance$AccPercentage <= 0.9, c('Vars', 'AccPercentage')]

## Vars AccPercentage  
## 1 marital.statusMarried-civ-spouse 0.1492073  
## 2 capital.gain 0.2869105  
## 3 age 0.3779174  
## 4 UnploymentRatio 0.4444315  
## 5 education.num 0.5060198  
## 6 hours.per.week 0.5650376  
## 7 fnlwgt 0.6162759  
## 8 marital.statusNever-married 0.6582691  
## 9 capital.loss 0.6990485  
## 10 relationshipNot-in-family 0.7271776  
## 11 occupationExec-managerial 0.7525710  
## 12 sexMale 0.7717234  
## 13 occupationProf-specialty 0.7864458  
## 14 relationshipUnmarried 0.8001563  
## 15 relationshipOwn-child 0.8134178  
## 16 educationBachelors 0.8265370  
## 17 educationMasters 0.8365606  
## 18 relationshipWife 0.8457519  
## 19 occupationOther-service 0.8546329  
## 20 workclassSelf-emp-not-inc 0.8623585  
## 21 workclassPrivate 0.8697457  
## 22 educationHS-grad 0.8766671  
## 23 workclassSelf-emp-inc 0.8831979  
## 24 educationProf-school 0.8891912  
## 25 native.countryUnited-States 0.8945208  
## 26 occupationSales 0.8995773

# representación gráfica  
plot(vImp, top =26)



Por tanto, eliminamos las variables raza y país de procedencia, ya que tienen una importancia residual en la disminución de la impureza. Además, eliminamos la variable fnlwgt por no tener interpretabilidad y la variable education.num, por ser una representación de la variable categórica y, por lo tanto, redundante.

# Eliminación de las variables redundantes o con ninguna capacidad predictiva  
census$education.num <- NULL  
census$race <- NULL  
census$native.country <- NULL  
census$fnlwgt <- NULL

# 6. Fusión de las categorías no significativas

Antes de proceder al entrenamiento de los algoritmos, haremos uso de las técnicas de clusterización mediante el uso de los árboles CHAID y CART, tal y como se ha mencionado anteriormente, para el tratamiento de los niveles de cada variable factorial y continua, respectivamente. El objetivo es reflejar en la base de datos los patrones detectados, resumiendo la información mediante el uso de principios estadísticos rigurosos y la eliminación, mediante el fusionado, de las categorías poco significativas. Para detectarlas, haremos uso del estadístico de Wald, implementado dentro de la familia de modelos generalizados, en particular, los de la familia binomial. Además, se comprobará con el estadístico de Wald que las categorías creadas son estadísticamente significativas.

## 6.1. Education

glm.model.education <- glm(y ~ education, family="binomial", data = census)  
summary(glm.model.education)

##   
## Call:  
## glm(formula = y ~ education, family = "binomial", data = census)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6409 -0.6486 -0.5877 -0.3228 2.9687   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.70575 0.11074 -24.434 <2e-16 \*\*\*  
## education11th -0.22254 0.15399 -1.445 0.1484   
## education12th 0.16677 0.18639 0.895 0.3709   
## education1st-4th -0.68287 0.37614 -1.815 0.0695 .   
## education5th-6th -0.17428 0.22667 -0.769 0.4420   
## education7th-8th 0.03942 0.17179 0.229 0.8185   
## education9th -0.15296 0.19507 -0.784 0.4330   
## educationAssoc-acdm 1.64917 0.12460 13.236 <2e-16 \*\*\*  
## educationAssoc-voc 1.62518 0.12177 13.346 <2e-16 \*\*\*  
## educationBachelors 2.35455 0.11303 20.831 <2e-16 \*\*\*  
## educationDoctorate 3.67811 0.14394 25.554 <2e-16 \*\*\*  
## educationHS-grad 1.03696 0.11286 9.188 <2e-16 \*\*\*  
## educationMasters 2.90368 0.11740 24.733 <2e-16 \*\*\*  
## educationPreschool -1.68870 1.01107 -1.670 0.0949 .   
## educationProf-school 3.75072 0.13598 27.582 <2e-16 \*\*\*  
## educationSome-college 1.25388 0.11341 11.057 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 53730 on 48812 degrees of freedom  
## Residual deviance: 47496 on 48797 degrees of freedom  
## AIC: 47528  
##   
## Number of Fisher Scoring iterations: 6

chaid.model <- chaid(y ~ education, data=census)  
chaid.model

##   
## Model formula:  
## y ~ education  
##   
## Fitted party:  
## [1] root  
## | [2] education in 10th, 11th, 12th, 1st-4th, 5th-6th, 7th-8th, 9th, Preschool: no (n = 6402, err = 5.7%)  
## | [3] education in Assoc-acdm, Assoc-voc: no (n = 3661, err = 25.5%)  
## | [4] education in Bachelors: no (n = 8020, err = 41.3%)  
## | [5] education in Doctorate, Prof-school: yes (n = 1428, err = 26.6%)  
## | [6] education in HS-grad: no (n = 15777, err = 15.9%)  
## | [7] education in Masters: yes (n = 2656, err = 45.1%)  
## | [8] education in Some-college: no (n = 10869, err = 19.0%)  
##   
## Number of inner nodes: 1  
## Number of terminal nodes: 7

Fusión de las categorías:

census <- census %>% mutate(education = case\_when(education == "1st-4th" ~ "Elemtary-education",  
 education == "5th-6th" ~ "Elemtary-education",  
 education == "7th-8th" ~ "Elemtary-education",  
 education == "9th" ~ "Elemtary-education",  
 education == "Preschool" ~ "Elemtary-education",  
 education == "10th" ~ "Elemtary-education",  
 education == "11th" ~ "Elemtary-education",  
 education == "12th" ~ "Elemtary-education",  
 education == "Assoc-acdm" ~ "Associate-degree",  
 education == "Assoc-voc" ~ "Associate-degree",  
 TRUE ~ as.character(education)) )  
census$education <- as.factor(census$education)  
  
  
glm.model.education <- glm(y ~ education, family="binomial", data = census)  
summary(glm.model.education)

##   
## Call:  
## glm(formula = y ~ education, family = "binomial", data = census)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6409 -0.6486 -0.5877 -0.3431 2.3924   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.07004 0.03790 -28.234 <2e-16 \*\*\*  
## educationBachelors 0.71885 0.04417 16.276 <2e-16 \*\*\*  
## educationDoctorate 2.04240 0.09946 20.536 <2e-16 \*\*\*  
## educationElemtary-education -1.73282 0.06583 -26.321 <2e-16 \*\*\*  
## educationHS-grad -0.59875 0.04372 -13.695 <2e-16 \*\*\*  
## educationMasters 1.26798 0.05438 23.317 <2e-16 \*\*\*  
## educationProf-school 2.11502 0.08755 24.157 <2e-16 \*\*\*  
## educationSome-college -0.38183 0.04511 -8.464 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 53730 on 48812 degrees of freedom  
## Residual deviance: 47510 on 48805 degrees of freedom  
## AIC: 47526  
##   
## Number of Fisher Scoring iterations: 5

## 6.2. Marital status

glm.model.marital.status <- glm(y ~ marital.status, family="binomial", data = census)  
summary(glm.model.marital.status)

##   
## Call:  
## glm(formula = y ~ marital.status, family = "binomial", data = census)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.0871 -1.0871 -0.3053 -0.3053 2.4857   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)  
## (Intercept) -2.18389 0.04072 -53.632 < 2e-16  
## marital.statusMarried-AF-spouse 1.68745 0.34142 4.943 7.71e-07  
## marital.statusMarried-civ-spouse 1.96778 0.04288 45.886 < 2e-16  
## marital.statusMarried-spouse-absent -0.10130 0.14371 -0.705 0.4809  
## marital.statusNever-married -0.85881 0.05556 -15.457 < 2e-16  
## marital.statusSeparated -0.48712 0.11161 -4.364 1.28e-05  
## marital.statusWidowed -0.20114 0.10095 -1.993 0.0463  
##   
## (Intercept) \*\*\*  
## marital.statusMarried-AF-spouse \*\*\*  
## marital.statusMarried-civ-spouse \*\*\*  
## marital.statusMarried-spouse-absent   
## marital.statusNever-married \*\*\*  
## marital.statusSeparated \*\*\*  
## marital.statusWidowed \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 53730 on 48812 degrees of freedom  
## Residual deviance: 43109 on 48806 degrees of freedom  
## AIC: 43123  
##   
## Number of Fisher Scoring iterations: 5

chaid.model <- chaid(y ~ marital.status, data=census)  
chaid.model

##   
## Model formula:  
## y ~ marital.status  
##   
## Fitted party:  
## [1] root  
## | [2] marital.status in Divorced: no (n = 6630, err = 10.1%)  
## | [3] marital.status in Married-AF-spouse, Married-civ-spouse: no (n = 22409, err = 44.6%)  
## | [4] marital.status in Married-spouse-absent, Widowed: no (n = 2146, err = 8.7%)  
## | [5] marital.status in Never-married: no (n = 16098, err = 4.6%)  
## | [6] marital.status in Separated: no (n = 1530, err = 6.5%)  
##   
## Number of inner nodes: 1  
## Number of terminal nodes: 5

census <- census %>% mutate(marital.status = case\_when(marital.status == "Married-AF-spouse" ~ "Married-AF-or-civ",  
 marital.status == "Married-civ-spouse" ~ "Married-AF-or-civ",  
 marital.status == "Married-spouse-absent" ~ "Spouse-absent-Widowed",  
 marital.status == "Widowed" ~ "Spouse-absent-Widowed",  
 TRUE ~ as.character(marital.status)) )  
census$marital.status <- as.factor(census$marital.status)  
  
  
glm.model.marital.status <- glm(y ~ marital.status, family="binomial", data = census)  
summary(glm.model.marital.status)

##   
## Call:  
## glm(formula = y ~ marital.status, family = "binomial", data = census)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.0869 -1.0869 -0.3053 -0.3053 2.4857   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)  
## (Intercept) -2.18389 0.04072 -53.632 < 2e-16  
## marital.statusMarried-AF-or-civ 1.96733 0.04288 45.879 < 2e-16  
## marital.statusNever-married -0.85881 0.05556 -15.457 < 2e-16  
## marital.statusSeparated -0.48712 0.11161 -4.364 1.28e-05  
## marital.statusSpouse-absent-Widowed -0.17106 0.08686 -1.969 0.0489  
##   
## (Intercept) \*\*\*  
## marital.statusMarried-AF-or-civ \*\*\*  
## marital.statusNever-married \*\*\*  
## marital.statusSeparated \*\*\*  
## marital.statusSpouse-absent-Widowed \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 53730 on 48812 degrees of freedom  
## Residual deviance: 43110 on 48808 degrees of freedom  
## AIC: 43120  
##   
## Number of Fisher Scoring iterations: 5

## 6.3. Relationship

En el caso de la relación, todas las categorías son significativas y no dan pie a la fusión de las mismas.

glm.relationship <- glm(y ~ relationship, family="binomial", data = census)  
summary(glm.relationship)

##   
## Call:  
## glm(formula = y ~ relationship, family = "binomial", data = census)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.1250 -0.4627 -0.3527 -0.1718 2.9063   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.20581 0.01432 -14.37 <2e-16 \*\*\*  
## relationshipNot-in-family -1.97447 0.03282 -60.15 <2e-16 \*\*\*  
## relationshipOther-relative -3.12502 0.14186 -22.03 <2e-16 \*\*\*  
## relationshipOwn-child -4.00264 0.09664 -41.42 <2e-16 \*\*\*  
## relationshipUnmarried -2.54034 0.06041 -42.05 <2e-16 \*\*\*  
## relationshipWife 0.08124 0.04391 1.85 0.0643 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 53730 on 48812 degrees of freedom  
## Residual deviance: 42537 on 48807 degrees of freedom  
## AIC: 42549  
##   
## Number of Fisher Scoring iterations: 6

chaid.model <- chaid(y ~ relationship, data=census)  
chaid.model

##   
## Model formula:  
## y ~ relationship  
##   
## Fitted party:  
## [1] root  
## | [2] relationship in Husband, Wife: no (n = 22040, err = 45.1%)  
## | [3] relationship in Not-in-family: no (n = 12567, err = 10.2%)  
## | [4] relationship in Other-relative: no (n = 1506, err = 3.5%)  
## | [5] relationship in Own-child: no (n = 7576, err = 1.5%)  
## | [6] relationship in Unmarried: no (n = 5124, err = 6.0%)  
##   
## Number of inner nodes: 1  
## Number of terminal nodes: 5

## 6.4. Occupation

glm.occupation<- glm(y ~ occupation, family="binomial", data = census)  
summary(glm.occupation)

##   
## Call:  
## glm(formula = y ~ occupation, family = "binomial", data = census)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.1257 -0.7685 -0.5312 -0.2838 2.9012   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.88718 0.03820 -49.399 < 2e-16 \*\*\*  
## occupationArmed-Forces 1.19404 0.54905 2.175 0.0297 \*   
## occupationCraft-repair 0.63140 0.04866 12.975 < 2e-16 \*\*\*  
## occupationExec-managerial 1.76428 0.04577 38.550 < 2e-16 \*\*\*  
## occupationFarming-fishing -0.18076 0.08886 -2.034 0.0419 \*   
## occupationHandlers-cleaners -0.78482 0.09398 -8.351 < 2e-16 \*\*\*  
## occupationMachine-op-inspct -0.11091 0.06646 -1.669 0.0951 .   
## occupationOther-service -1.30447 0.07924 -16.462 < 2e-16 \*\*\*  
## occupationPriv-house-serv -2.30625 0.50500 -4.567 4.95e-06 \*\*\*  
## occupationProf-specialty 1.66046 0.04573 36.311 < 2e-16 \*\*\*  
## occupationProtective-serv 1.04868 0.07797 13.449 < 2e-16 \*\*\*  
## occupationSales 0.81864 0.04848 16.886 < 2e-16 \*\*\*  
## occupationTech-support 0.95429 0.06856 13.920 < 2e-16 \*\*\*  
## occupationTransport-moving 0.49281 0.06317 7.801 6.15e-15 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 53730 on 48812 degrees of freedom  
## Residual deviance: 47622 on 48799 degrees of freedom  
## AIC: 47650  
##   
## Number of Fisher Scoring iterations: 6

chaid.occupation <- chaid(y ~ occupation, data=census)  
chaid.occupation

##   
## Model formula:  
## y ~ occupation  
##   
## Fitted party:  
## [1] root  
## | [2] occupation in ?, Adm-clerical: no (n = 5997, err = 13.2%)  
## | [3] occupation in Armed-Forces, Protective-serv: no (n = 1042, err = 30.2%)  
## | [4] occupation in Craft-repair: no (n = 6378, err = 22.2%)  
## | [5] occupation in Exec-managerial, Machine-op-inspct  
## | | [6] occupation in ?, Adm-clerical, Armed-Forces, Craft-repair, Exec-managerial, Farming-fishing, Handlers-cleaners, Other-service, Priv-house-serv, Prof-specialty, Protective-serv, Sales, Tech-support, Transport-moving: no (n = 6322, err = 46.9%)  
## | | [7] occupation in Machine-op-inspct: no (n = 3216, err = 11.9%)  
## | [8] occupation in Farming-fishing: no (n = 1559, err = 11.2%)  
## | [9] occupation in Handlers-cleaners: no (n = 2243, err = 6.5%)  
## | [10] occupation in Other-service: no (n = 5471, err = 3.9%)  
## | [11] occupation in Priv-house-serv: no (n = 269, err = 1.5%)  
## | [12] occupation in Prof-specialty: no (n = 6414, err = 44.4%)  
## | [13] occupation in Sales: no (n = 5898, err = 25.6%)  
## | [14] occupation in Tech-support: no (n = 1523, err = 28.2%)  
## | [15] occupation in Transport-moving: no (n = 2481, err = 19.9%)  
##   
## Number of inner nodes: 2  
## Number of terminal nodes: 13

census <- census %>% mutate(occupation = case\_when(occupation == "Armed-Forces" ~ "Armed-Forces-Protective-serv",  
 occupation == "Protective-serv" ~ "Armed-Forces-Protective-serv",  
 occupation == "Exec-managerial" ~ "Exec-managerial-Machine-op-inspct",  
 occupation == "Machine-op-inspct" ~ "Exec-managerial-Machine-op-inspct",  
 TRUE ~ as.character(occupation))  
 )  
census$occupation <- as.factor(census$occupation)  
  
glm.occupation<- glm(y ~ occupation, family="binomial", data = census)  
summary(glm.occupation)

##   
## Call:  
## glm(formula = y ~ occupation, family = "binomial", data = census)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.0828 -0.7685 -0.5312 -0.2838 2.9012   
##   
## Coefficients:  
## Estimate Std. Error z value  
## (Intercept) -1.88718 0.03820 -49.399  
## occupationArmed-Forces-Protective-serv 1.05083 0.07752 13.555  
## occupationCraft-repair 0.63140 0.04866 12.975  
## occupationExec-managerial-Machine-op-inspct 1.27399 0.04381 29.079  
## occupationFarming-fishing -0.18076 0.08886 -2.034  
## occupationHandlers-cleaners -0.78482 0.09398 -8.351  
## occupationOther-service -1.30447 0.07924 -16.462  
## occupationPriv-house-serv -2.30625 0.50500 -4.567  
## occupationProf-specialty 1.66046 0.04573 36.311  
## occupationSales 0.81864 0.04848 16.886  
## occupationTech-support 0.95429 0.06856 13.920  
## occupationTransport-moving 0.49281 0.06317 7.801  
## Pr(>|z|)   
## (Intercept) < 2e-16 \*\*\*  
## occupationArmed-Forces-Protective-serv < 2e-16 \*\*\*  
## occupationCraft-repair < 2e-16 \*\*\*  
## occupationExec-managerial-Machine-op-inspct < 2e-16 \*\*\*  
## occupationFarming-fishing 0.0419 \*   
## occupationHandlers-cleaners < 2e-16 \*\*\*  
## occupationOther-service < 2e-16 \*\*\*  
## occupationPriv-house-serv 4.95e-06 \*\*\*  
## occupationProf-specialty < 2e-16 \*\*\*  
## occupationSales < 2e-16 \*\*\*  
## occupationTech-support < 2e-16 \*\*\*  
## occupationTransport-moving 6.15e-15 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 53730 on 48812 degrees of freedom  
## Residual deviance: 48895 on 48801 degrees of freedom  
## AIC: 48919  
##   
## Number of Fisher Scoring iterations: 6

## 6.5. workclass

glm.workclass <- glm(y ~ workclass, family="binomial", data = census)  
summary(glm.workclass)

##   
## Call:  
## glm(formula = y ~ workclass, family = "binomial", data = census)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.270 -0.684 -0.684 -0.684 2.169   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.43992 0.05414 -8.126 4.43e-16 \*\*\*  
## workclassLocal-gov -0.42842 0.06680 -6.414 1.42e-10 \*\*\*  
## workclassNever-worked -11.12613 62.28667 -0.179 0.8582   
## workclassPrivate -0.89363 0.05564 -16.061 < 2e-16 \*\*\*  
## workclassSelf-emp-inc 0.65563 0.07293 8.989 < 2e-16 \*\*\*  
## workclassSelf-emp-not-inc -0.50979 0.06495 -7.849 4.19e-15 \*\*\*  
## workclassState-gov -0.56721 0.07421 -7.644 2.11e-14 \*\*\*  
## workclassWithout-pay -1.81137 0.74536 -2.430 0.0151 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 53730 on 48812 degrees of freedom  
## Residual deviance: 52502 on 48805 degrees of freedom  
## AIC: 52518  
##   
## Number of Fisher Scoring iterations: 10

chaid.workclass <- chaid(y ~ workclass, data=census)  
chaid.workclass

##   
## Model formula:  
## y ~ workclass  
##   
## Fitted party:  
## [1] root  
## | [2] workclass in ?, Federal-gov, Self-emp-inc, Self-emp-not-inc, State-gov  
## | | [3] workclass in ?, Federal-gov, Local-gov, Never-worked, Private, Self-emp-inc, State-gov, Without-pay  
## | | | [4] workclass in ?, Federal-gov, Local-gov, Never-worked, Private, Self-emp-not-inc, Without-pay: no (n = 1432, err = 39.2%)  
## | | | [5] workclass in Self-emp-inc: yes (n = 1694, err = 44.6%)  
## | | | [6] workclass in State-gov: no (n = 1981, err = 26.8%)  
## | | [7] workclass in Self-emp-not-inc: no (n = 3861, err = 27.9%)  
## | [8] workclass in Local-gov: no (n = 3136, err = 29.6%)  
## | [9] workclass in Never-worked, Without-pay: no (n = 31, err = 6.5%)  
## | [10] workclass in Private: no (n = 36678, err = 20.9%)  
##   
## Number of inner nodes: 3  
## Number of terminal nodes: 7

census <- census %>% mutate(workclass = case\_when(workclass == "Never-worked" ~ "Never-worked-No-Pay",  
 workclass == "Without-pay" ~ "Never-worked-No-Pay",  
 TRUE ~ as.character(workclass))  
 )  
census$workclass <- as.factor(census$workclass)  
  
  
glm.workclass <- glm(y ~ workclass, family="binomial", data = census)  
summary(glm.workclass)

##   
## Call:  
## glm(formula = y ~ workclass, family = "binomial", data = census)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.270 -0.684 -0.684 -0.684 2.341   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.43992 0.05414 -8.126 4.43e-16 \*\*\*  
## workclassLocal-gov -0.42842 0.06680 -6.414 1.42e-10 \*\*\*  
## workclassNever-worked-No-Pay -2.23422 0.73151 -3.054 0.00226 \*\*   
## workclassPrivate -0.89363 0.05564 -16.061 < 2e-16 \*\*\*  
## workclassSelf-emp-inc 0.65563 0.07293 8.989 < 2e-16 \*\*\*  
## workclassSelf-emp-not-inc -0.50979 0.06495 -7.849 4.19e-15 \*\*\*  
## workclassState-gov -0.56721 0.07421 -7.644 2.11e-14 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 53730 on 48812 degrees of freedom  
## Residual deviance: 52504 on 48806 degrees of freedom  
## AIC: 52518  
##   
## Number of Fisher Scoring iterations: 4

## 6.6. capital.gain

Como se intuía en el análisis gráfico, existe un punto de corte relevante, que discrimina muy bien las clases, por lo que incorporamos este conocimiento en la representación de la base de datos.

# Cálculo del punto de corte óptim usando el algoritmo CART  
model.rpart <- rpart(y ~ capital.gain, data=census, method = "class")  
model.rpart

## n= 48813   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 48813 11685 no (0.76061705 0.23938295)   
## 2) capital.gain< 5119 46481 9477 no (0.79611024 0.20388976) \*  
## 3) capital.gain>=5119 2332 124 yes (0.05317324 0.94682676) \*

census <- census %>% mutate(capital.gain.f = case\_when(capital.gain >= 5119 ~ "HighCapitalGain",  
 capital.gain < 5119 ~ "ModerateCapitalGain")   
 )  
census$capital.gain.f <- as.factor(census$capital.gain.f)  
  
# Validación de los nuevos factores  
glm.capital.gain <- glm(y ~ capital.gain.f, family="binomial", data = census)  
summary(glm.capital.gain)

##   
## Call:  
## glm(formula = y ~ capital.gain.f, family = "binomial", data = census)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.4225 -0.6753 -0.6753 -0.6753 1.7833   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 2.87956 0.09229 31.20 <2e-16 \*\*\*  
## capital.gain.fModerateCapitalGain -4.24172 0.09300 -45.61 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 53730 on 48812 degrees of freedom  
## Residual deviance: 47984 on 48811 degrees of freedom  
## AIC: 47988  
##   
## Number of Fisher Scoring iterations: 5

census$capital.gain <- census$capital.gain.f  
census$capital.gain.f <- NULL

## 6.7. capital.loss

Se muestran los cortes óptimos hallados por el árbol CART. Aunque existe la posibilidad de discretizar la variable, no lo vamos a hacer, ya que en este caso no están tan claros los puntos de corte óptimos y, por lo tanto, delegaremos la tarea en los distintos algoritmos.

# Cálculo del punto de corte óptim usando el algoritmo CART  
model.rpart <- rpart(y ~ capital.loss, data=census, method = "class")  
model.rpart

## n= 48813   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 48813 11685 no (0.7606170 0.2393830)   
## 2) capital.loss< 1820.5 47329 10634 no (0.7753175 0.2246825) \*  
## 3) capital.loss>=1820.5 1484 433 yes (0.2917790 0.7082210)   
## 6) capital.loss>=1978.5 527 206 no (0.6091082 0.3908918)   
## 12) capital.loss< 2364.5 324 44 no (0.8641975 0.1358025) \*  
## 13) capital.loss>=2364.5 203 41 yes (0.2019704 0.7980296) \*  
## 7) capital.loss< 1978.5 957 112 yes (0.1170324 0.8829676) \*

# 7. Discretización de variables y equilibrado de la muestra

Si observamos los resultados del modelo random forest entrenado para la selección de variables:

rf\_sv.fit

## Parallel Random Forest   
##   
## 48813 samples  
## 15 predictor  
## 2 classes: 'no', 'yes'   
##   
## Pre-processing: centered (101), scaled (101)   
## Resampling: Cross-Validated (10 fold, repeated 5 times)   
## Summary of sample sizes: 43932, 43931, 43931, 43933, 43931, 43932, ...   
## Resampling results across tuning parameters:  
##   
## mtry ROC Sens Spec Accuracy Kappa   
## 5 0.8855485 0.9510289 0.5631329 0.8581731 0.5689460  
## 6 0.8903577 0.9484270 0.5823367 0.8607912 0.5812505  
## 7 0.8945289 0.9460569 0.5966457 0.8624138 0.5894868  
## 8 0.8975453 0.9451250 0.6067102 0.8641141 0.5965233  
## 9 0.9004707 0.9439991 0.6146690 0.8651630 0.6013444  
## 10 0.9028594 0.9432127 0.6172537 0.8651836 0.6021586  
##   
## ROC was used to select the optimal model using the largest value.  
## The final value used for the model was mtry = 10.

Observamos que se da una predicción notablemente mejor en la clase mayoritaria, ilustrando el problema de desbalanceo en los problemas de clasificación, en donde la clase minoritaría se ve perjudicada. En este caso particular, el efecto es muy evidente, siendo las predicciones sobre la clase minoritaría cercanas a las que daría un predictor totalmente aleatorio. Así, se hace evidente la necesidad de balancear las clases de cara al entrenamiento de los algoritmos. Igualmente, codificamos los niveles de las variables categóricas en variables dummy.

# introducción de variables dummy: una variable por cada nivel de cada factor  
census.numeric <- dummy.data.frame(census[, c('education', 'marital.status', 'occupation', 'relationship', 'sex', 'capital.gain', 'capital.loss', 'workclass')])

# Añadimos las variables continuas  
census.numeric$UnploymentRatio <- census$UnploymentRatio  
census.numeric$age <- census$age  
census.numeric$hours.per.week <- census$hours.per.week  
census.numeric$y <- census$y  
  
# Cálculo de los parámetros para el método del cubo  
datos\_gt50K <- subset(census.numeric, census.numeric$y == "yes")  
datos\_lt50K <- subset(census.numeric, census.numeric$y == "no")  
  
nA <- nrow(datos\_lt50K)  
nB <- nrow(datos\_gt50K)  
  
UNO=rep(1,dim(datos\_lt50K)[1])  
  
pik = rep(nB/nA, nA)  
  
X <- datos\_lt50K[, names(datos\_lt50K) != "y"]  
  
UNO <- rep( 1, dim( datos\_lt50K )[ 1 ])  
  
X <- as.matrix( cbind( UNO, X ) )  
  
# Método del cubo  
s=samplecube(X, pik, method=2, order = 1, comment = TRUE)  
## HorvitzThompson\_estimators  
## UNO 3.712800e+04  
## educationAssociate-degree 2.726215e+03  
## educationBachelors 4.708917e+03  
## educationDoctorate 1.620478e+02  
## educationElemtary-education 6.033896e+03  
## educationHS-grad 1.327521e+04  
## educationMasters 1.197882e+03  
## educationProf-school 2.160637e+02  
## educationSome-college 8.807772e+03  
## marital.statusDivorced 5.957638e+03  
## marital.statusMarried-AF-or-civ 1.241095e+04  
## marital.statusNever-married 1.536594e+04  
## marital.statusSeparated 1.433011e+03  
## marital.statusSpouse-absent-Widowed 1.960460e+03  
## occupationAdm-clerical 5.210947e+03  
## occupationArmed-Forces-Protective-serv 7.244488e+02  
## occupationCraft-repair 4.963110e+03  
## occupationExec-managerial-Machine-op-inspct 6.183234e+03  
## occupationFarming-fishing 1.385349e+03  
## occupationHandlers-cleaners 2.097089e+03  
## occupationOther-service 5.255431e+03  
## occupationPriv-house-serv 2.669022e+02  
## occupationProf-specialty 3.568228e+03  
## occupationSales 4.391176e+03  
## occupationTech-support 1.096205e+03  
## occupationTransport-moving 1.985879e+03  
## relationshipHusband 1.086355e+04  
## relationshipNot-in-family 1.129250e+04  
## relationshipOther-relative 1.452075e+03  
## relationshipOwn-child 7.466906e+03  
## relationshipUnmarried 4.816949e+03  
## relationshipWife 1.236011e+03  
## sexFemale 1.441272e+04  
## sexMale 2.271528e+04  
## capital.gainHighCapitalGain 1.239189e+02  
## capital.gainModerateCapitalGain 3.700408e+04  
## capital.loss 2.016602e+06  
## workclassFederal-gov 8.737869e+02  
## workclassLocal-gov 2.205120e+03  
## workclassNever-worked-No-Pay 2.859666e+01  
## workclassPrivate 2.902879e+04  
## workclassSelf-emp-inc 7.562228e+02  
## workclassSelf-emp-not-inc 2.783408e+03  
## workclassState-gov 1.452075e+03  
## UnploymentRatio 1.725129e+05  
## age 1.369062e+06  
## hours.per.week 1.442111e+06  
## Relative\_deviation  
## UNO 2.122350e-11  
## educationAssociate-degree 7.892429e-03  
## educationBachelors 4.072813e-02  
## educationDoctorate -5.842003e-01  
## educationElemtary-education -3.486143e-02  
## educationHS-grad 1.553007e-03  
## educationMasters 7.371874e-02  
## educationProf-school -4.314878e-01  
## educationSome-college 8.765929e-03  
## marital.statusDivorced -2.285623e-02  
## marital.statusMarried-AF-or-civ -1.650305e-02  
## marital.statusNever-married 6.117299e-03  
## marital.statusSeparated 1.404980e-01  
## marital.statusSpouse-absent-Widowed 2.347332e-02  
## occupationAdm-clerical 5.659310e-02  
## occupationArmed-Forces-Protective-serv -3.509243e-01  
## occupationCraft-repair -1.793659e-02  
## occupationExec-managerial-Machine-op-inspct -6.087135e-02  
## occupationFarming-fishing 9.750161e-02  
## occupationHandlers-cleaners -4.344256e-02  
## occupationOther-service 8.202958e-03  
## occupationPriv-house-serv 7.178046e-01  
## occupationProf-specialty -2.163114e-02  
## occupationSales 2.679681e-02  
## occupationTech-support 2.932655e-01  
## occupationTransport-moving -1.066734e-01  
## relationshipHusband -1.330602e-02  
## relationshipNot-in-family 1.332244e-02  
## relationshipOther-relative -1.323956e-01  
## relationshipOwn-child 2.553637e-02  
## relationshipUnmarried 4.047578e-02  
## relationshipWife -1.606384e-01  
## sexFemale -1.957654e-03  
## sexMale 1.242160e-03  
## capital.gainHighCapitalGain -6.542714e-02  
## capital.gainModerateCapitalGain 2.192457e-04  
## capital.loss 2.279645e-01  
## workclassFederal-gov 3.199663e-01  
## workclassLocal-gov -1.756265e-01  
## workclassNever-worked-No-Pay -1.390819e+00  
## workclassPrivate 2.720582e-03  
## workclassSelf-emp-inc 2.947749e-02  
## workclassSelf-emp-not-inc -2.124740e-02  
## workclassState-gov 7.408462e-02  
## UnploymentRatio -1.993434e-03  
## age -4.891109e-03  
## hours.per.week -2.591983e-03

# Equilibrado de la muestra  
muestra = cbind(datos\_lt50K,s)  
muestra1 <- muestra[muestra$s == 1,]  
muestra1$s <- NULL  
datos\_balanceados <- rbind(muestra1,datos\_gt50K)

# 8. Entrenamiento de los algoritmos

Los algoritmos seleccionados para el entrenamiento son:

* CART
* Random Forest
* C5.0
* KNN
* Redes neunorales
* Xgboost

Fundamentalmente, se ha optado por los algoritmos tipo árbol, ya que favorecen notablemente la interpretación de los modelos. De igual manera, se han elegido algoritmos con otro tipo de estructura interna, especialmente destacables los de la familia gradient boosting.

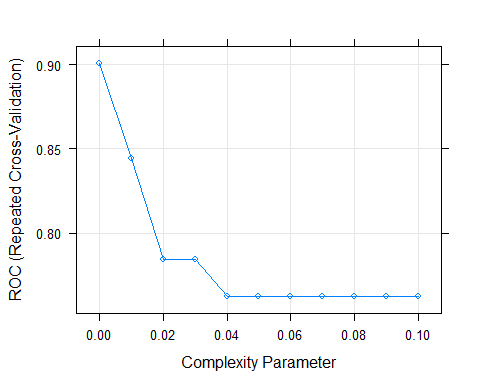
# Se cambia el nombre de las columnas a un formato compatible con el entrenamiento de los modelos  
colnames(datos\_balanceados) <- make.names(colnames(datos\_balanceados))  
  
fiveStats = function(...) c (twoClassSummary(...), defaultSummary(...))  
train.control <- trainControl(method = "repeatedcv", number = 10, repeats = 5, classProbs = TRUE, summaryFunction = fiveStats)  
  
# Creación de datos de entrenamiento y validación  
train.index <- createDataPartition(y = datos\_balanceados$y, p= 0.8, list = FALSE)  
data.training <- datos\_balanceados[train.index,]  
data.testing <- datos\_balanceados[-train.index,]

get\_model\_results <- function(model, data\_training, data\_testing, plot\_param, show\_confusionMatrix) {  
 print(paste("#######################################################"))  
 print(paste("Matriz de confusión del modelo:"))  
 print(paste("#######################################################"))  
 print(confusionMatrix(model))  
 if(show\_confusionMatrix) {  
 print(paste("#######################################################"))  
 print(paste("Matriz de confusión comparando las predicciones del modelo y los datos de test:"))  
 print(paste("#######################################################"))  
 print(confusionMatrix(predict(model, newdata = data\_testing), data\_testing$y))  
 }  
 print(paste("#######################################################"))  
 print(paste("Importancia de las variables:"))  
 print(varImp(model))  
 if(plot\_param) {  
 print(paste("Gráfico ROC-parámetros:"))  
 plot(model)  
 }  
}  
  
   
plot\_roc\_curve <- function(model, data\_training, data\_testing ) {  
 print("Curva ROC:")  
 pred\_prob\_val <- predict(model, data\_testing, type="prob")  
 pred\_prob\_ent <- predict(model, data\_training, type="prob")  
   
   
 curvaROC\_val <- roc(data\_testing$y,pred\_prob\_val[,"yes"])  
 curvaROC\_ent <- roc(data\_training$y,pred\_prob\_ent[,"yes"])  
 print(paste("Curva ROC para los datos de entrenamiento:"))  
 print(curvaROC\_ent)  
 print(paste("Curva ROC para los datos de validación:"))  
 print(curvaROC\_val)  
   
 plot(curvaROC\_ent,col="blue", main="Simulación con la curva ROC del modelo")  
 plot(curvaROC\_val, col="red", add=TRUE)  
 legend("bottomright", legend = c("Entrenamiento", "Validacion"), col = c("blue", "red"), lwd = 2)  
}

## 8.1. CART

#set.seed(999)  
#rpartfGrid <- expand.grid(cp = (0:10)/100)  
#clusterCPU <- registerDoParallel(cores=detectCores() - 1)  
#rpart.fit <- train(y ~ .,   
# data = data.training,   
# method = "rpart",  
# preProc = c("center", "scale"),   
# trControl = train.control,  
# metric = "ROC",  
# tuneGrid = rpartfGrid  
# )  
#stopImplicitCluster()  
#rpart.fit  
#saveRDS(rpart.fit, file = "rpart.rds")  
  
rpart.fit <- readRDS(file = "rpart.rds")  
get\_model\_results(rpart.fit, data.training, data.testing, TRUE, TRUE)

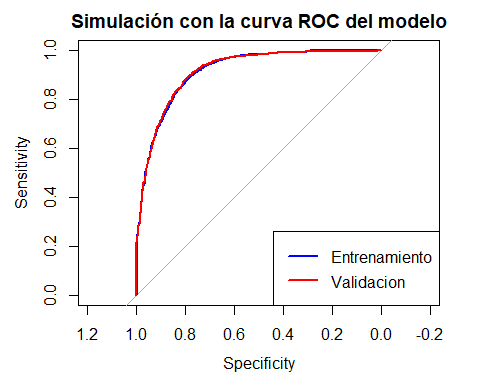
## [1] "#######################################################"  
## [1] "Matriz de confusión del modelo:"  
## [1] "#######################################################"  
## Cross-Validated (10 fold, repeated 5 times) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction no yes  
## no 39.7 8.8  
## yes 10.3 41.2  
##   
## Accuracy (average) : 0.8096  
##   
## [1] "#######################################################"  
## [1] "Matriz de confusión comparando las predicciones del modelo y los datos de test:"  
## [1] "#######################################################"  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 1886 304  
## yes 451 2033  
##   
## Accuracy : 0.8385   
## 95% CI : (0.8276, 0.8489)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6769   
##   
## Mcnemar's Test P-Value : 1.075e-07   
##   
## Sensitivity : 0.8070   
## Specificity : 0.8699   
## Pos Pred Value : 0.8612   
## Neg Pred Value : 0.8184   
## Prevalence : 0.5000   
## Detection Rate : 0.4035   
## Detection Prevalence : 0.4685   
## Balanced Accuracy : 0.8385   
##   
## 'Positive' Class : no   
##   
## [1] "#######################################################"  
## [1] "Importancia de las variables:"  
## rpart variable importance  
##   
## only 20 most important variables shown (out of 46)  
##   
## Overall  
## age 100.000  
## marital.statusMarried.AF.or.civ 90.280  
## UnploymentRatio 76.364  
## relationshipHusband 70.863  
## marital.statusNever.married 58.198  
## hours.per.week 40.363  
## capital.gainModerateCapitalGain 33.421  
## capital.gainHighCapitalGain 33.421  
## capital.loss 19.199  
## educationElemtary.education 16.765  
## occupationOther.service 7.586  
## educationHS.grad 6.666  
## occupationProf.specialty 4.919  
## occupationFarming.fishing 4.420  
## sexFemale 4.036  
## occupationExec.managerial.Machine.op.inspct 3.768  
## sexMale 3.564  
## educationBachelors 3.475  
## workclassPrivate 3.432  
## educationSome.college 3.219  
## [1] "Gráfico ROC-parámetros:"



plot\_roc\_curve(rpart.fit, data.training, data.testing)

## [1] "Curva ROC:"

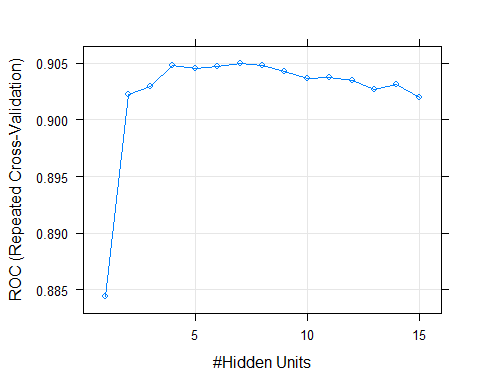
## [1] "Curva ROC para los datos de entrenamiento:"  
## Area under the curve: 0.9144  
## [1] "Curva ROC para los datos de validación:"  
## Area under the curve: 0.9155



## 8.2. Redes neuronales

#set.seed(999)  
#nnGrid <- expand.grid(size = 1:15)  
#clusterCPU <- registerDoParallel(cores=detectCores() - 1)  
#nn.fit <- train(y ~ .,   
# data = data.training,   
# method = "mlp",  
# preProc = c("center", "scale"),   
# trControl = train.control,  
# metric = "ROC",  
# tuneGrid = nnGrid  
# )  
#stopImplicitCluster()  
#nn.fit  
#saveRDS(nn.fit, file = "nn.fit.rds")  
  
nn.fit <- readRDS(file = "nn.fit.rds")  
get\_model\_results(nn.fit, data.training, data.testing, TRUE, TRUE)

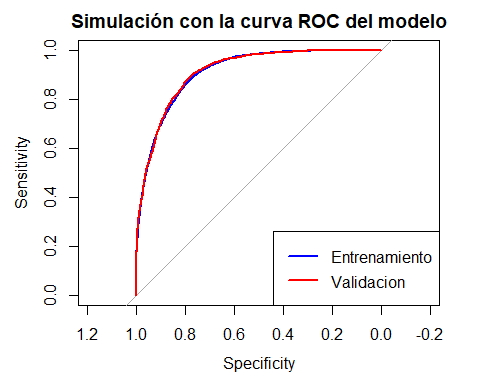
## [1] "#######################################################"  
## [1] "Matriz de confusión del modelo:"  
## [1] "#######################################################"  
## Cross-Validated (10 fold, repeated 5 times) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction no yes  
## no 39.9 7.8  
## yes 10.1 42.2  
##   
## Accuracy (average) : 0.8205  
##   
## [1] "#######################################################"  
## [1] "Matriz de confusión comparando las predicciones del modelo y los datos de test:"  
## [1] "#######################################################"  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 1852 281  
## yes 485 2056  
##   
## Accuracy : 0.8361   
## 95% CI : (0.8252, 0.8466)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6722   
##   
## Mcnemar's Test P-Value : 2.222e-13   
##   
## Sensitivity : 0.7925   
## Specificity : 0.8798   
## Pos Pred Value : 0.8683   
## Neg Pred Value : 0.8091   
## Prevalence : 0.5000   
## Detection Rate : 0.3962   
## Detection Prevalence : 0.4564   
## Balanced Accuracy : 0.8361   
##   
## 'Positive' Class : no   
##   
## [1] "#######################################################"  
## [1] "Importancia de las variables:"  
## ROC curve variable importance  
##   
## only 20 most important variables shown (out of 46)  
##   
## Importance  
## marital.statusMarried.AF.or.civ 100.00  
## relationshipHusband 89.23  
## UnploymentRatio 74.16  
## age 69.77  
## marital.statusNever.married 67.84  
## hours.per.week 65.85  
## sexFemale 45.36  
## sexMale 45.36  
## relationshipNot.in.family 37.66  
## relationshipOwn.child 35.92  
## capital.gainModerateCapitalGain 35.15  
## capital.gainHighCapitalGain 35.15  
## educationBachelors 29.81  
## occupationProf.specialty 29.08  
## educationHS.grad 28.46  
## educationElemtary.education 24.56  
## workclassPrivate 23.93  
## occupationOther.service 23.23  
## occupationExec.managerial.Machine.op.inspct 23.05  
## relationshipUnmarried 20.15  
## [1] "Gráfico ROC-parámetros:"



plot\_roc\_curve(nn.fit, data.training, data.testing)

## [1] "Curva ROC:"

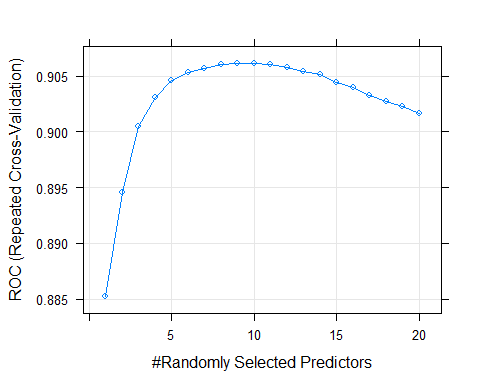
## [1] "Curva ROC para los datos de entrenamiento:"  
## Area under the curve: 0.9105  
## [1] "Curva ROC para los datos de validación:"  
## Area under the curve: 0.9126



## 8.3. Random Forest

#set.seed(999)  
#rfGrid <- expand.grid(mtry = 1:20)  
#clusterCPU <- registerDoParallel(cores=detectCores() - 1)  
#rf.fit <- train(y ~ .,   
# data = data.training,   
# method = "parRF",  
# preProc = c("center", "scale"),   
# trControl = train.control,  
# metric = "ROC",  
# tuneGrid = rfGrid  
# )  
#stopImplicitCluster()  
#rf.fit  
#saveRDS(rf.fit, file = "rf.fit.rds")  
  
rf.fit <- readRDS(file = "rf.fit.rds")  
get\_model\_results(rf.fit, data.training, data.testing, TRUE, TRUE)

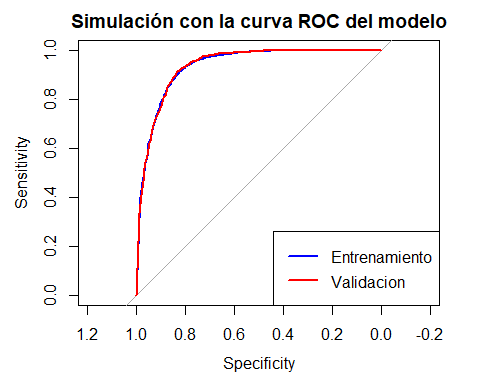
## [1] "#######################################################"  
## [1] "Matriz de confusión del modelo:"  
## [1] "#######################################################"  
## Cross-Validated (10 fold, repeated 5 times) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction no yes  
## no 39.8 6.9  
## yes 10.2 43.1  
##   
## Accuracy (average) : 0.8289  
##   
## [1] "#######################################################"  
## [1] "Matriz de confusión comparando las predicciones del modelo y los datos de test:"  
## [1] "#######################################################"  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 1898 175  
## yes 439 2162  
##   
## Accuracy : 0.8686   
## 95% CI : (0.8586, 0.8782)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7373   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.8122   
## Specificity : 0.9251   
## Pos Pred Value : 0.9156   
## Neg Pred Value : 0.8312   
## Prevalence : 0.5000   
## Detection Rate : 0.4061   
## Detection Prevalence : 0.4435   
## Balanced Accuracy : 0.8686   
##   
## 'Positive' Class : no   
##   
## [1] "#######################################################"  
## [1] "Importancia de las variables:"  
## parRF variable importance  
##   
## only 20 most important variables shown (out of 46)  
##   
## Overall  
## marital.statusMarried.AF.or.civ 100.000  
## age 87.696  
## relationshipHusband 46.234  
## UnploymentRatio 45.400  
## hours.per.week 45.277  
## capital.gainModerateCapitalGain 28.339  
## capital.gainHighCapitalGain 23.291  
## marital.statusNever.married 22.919  
## capital.loss 17.089  
## educationElemtary.education 12.640  
## relationshipWife 9.750  
## occupationProf.specialty 7.571  
## relationshipNot.in.family 6.917  
## relationshipOwn.child 6.816  
## occupationExec.managerial.Machine.op.inspct 6.197  
## workclassPrivate 6.120  
## occupationOther.service 5.954  
## educationHS.grad 5.941  
## educationBachelors 5.888  
## sexFemale 5.374  
## [1] "Gráfico ROC-parámetros:"



plot\_roc\_curve(rf.fit, data.training, data.testing)

## [1] "Curva ROC:"

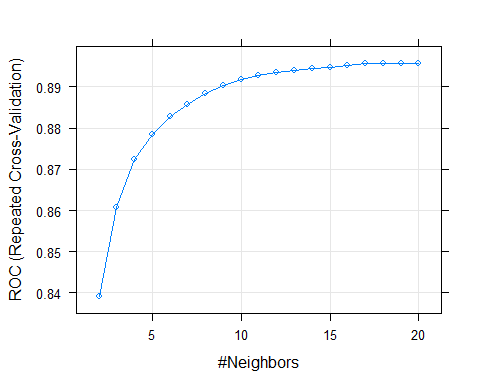
## [1] "Curva ROC para los datos de entrenamiento:"  
## Area under the curve: 0.9353  
## [1] "Curva ROC para los datos de validación:"  
## Area under the curve: 0.9354



## 8.4. Vecimos más próximos

#set.seed(999)  
#knnGrid <- expand.grid(k = 2:20)  
#clusterCPU <- registerDoParallel(cores=detectCores() - 1)  
#knn.fit <- train(y ~ .,   
# data = data.training,   
# method = "knn",  
# preProc = c("center", "scale"),   
# trControl = train.control,  
# metric = "ROC",  
# tuneGrid = knnGrid  
# )  
#stopImplicitCluster()  
#knn.fit  
#saveRDS(knn.fit, file = "knn.fit.rds")  
  
knn.fit <- readRDS(file = "knn.fit.rds")  
get\_model\_results(knn.fit, data.training, data.testing, TRUE, TRUE)

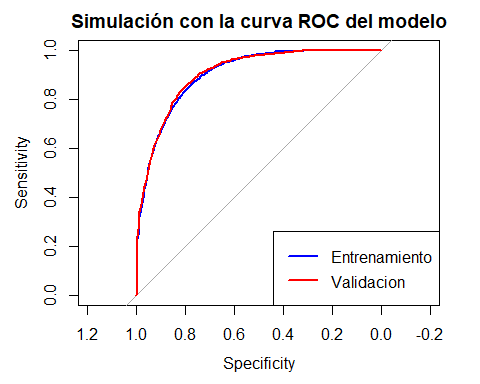
## [1] "#######################################################"  
## [1] "Matriz de confusión del modelo:"  
## [1] "#######################################################"  
## Cross-Validated (10 fold, repeated 5 times) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction no yes  
## no 39.1 7.7  
## yes 10.9 42.3  
##   
## Accuracy (average) : 0.8141  
##   
## [1] "#######################################################"  
## [1] "Matriz de confusión comparando las predicciones del modelo y los datos de test:"  
## [1] "#######################################################"  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 1836 317  
## yes 501 2020  
##   
## Accuracy : 0.825   
## 95% CI : (0.8138, 0.8358)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.65   
##   
## Mcnemar's Test P-Value : 1.57e-10   
##   
## Sensitivity : 0.7856   
## Specificity : 0.8644   
## Pos Pred Value : 0.8528   
## Neg Pred Value : 0.8013   
## Prevalence : 0.5000   
## Detection Rate : 0.3928   
## Detection Prevalence : 0.4606   
## Balanced Accuracy : 0.8250   
##   
## 'Positive' Class : no   
##   
## [1] "#######################################################"  
## [1] "Importancia de las variables:"  
## ROC curve variable importance  
##   
## only 20 most important variables shown (out of 46)  
##   
## Importance  
## marital.statusMarried.AF.or.civ 100.00  
## relationshipHusband 89.23  
## UnploymentRatio 74.16  
## age 69.77  
## marital.statusNever.married 67.84  
## hours.per.week 65.85  
## sexMale 45.36  
## sexFemale 45.36  
## relationshipNot.in.family 37.66  
## relationshipOwn.child 35.92  
## capital.gainHighCapitalGain 35.15  
## capital.gainModerateCapitalGain 35.15  
## educationBachelors 29.81  
## occupationProf.specialty 29.08  
## educationHS.grad 28.46  
## educationElemtary.education 24.56  
## workclassPrivate 23.93  
## occupationOther.service 23.23  
## occupationExec.managerial.Machine.op.inspct 23.05  
## relationshipUnmarried 20.15  
## [1] "Gráfico ROC-parámetros:"



plot\_roc\_curve(knn.fit, data.training, data.testing)

## [1] "Curva ROC:"

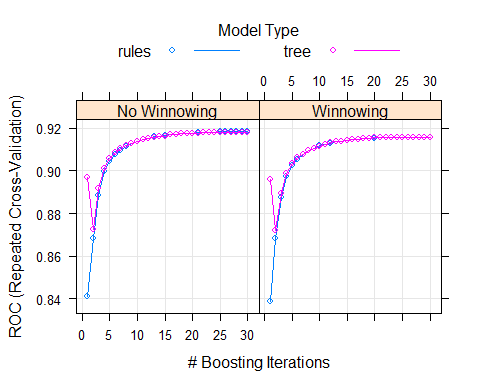
## [1] "Curva ROC para los datos de entrenamiento:"  
## Area under the curve: 0.9026  
## [1] "Curva ROC para los datos de validación:"  
## Area under the curve: 0.9055



## 8.5. C5.0

#set.seed(999)  
  
#c5Grid <- expand.grid(trials = 1:30,   
# model = c("tree", "rules"),   
# winnow = c(TRUE, FALSE))  
#clusterCPU <- registerDoParallel(cores=detectCores() - 1)  
#c5\_0.fit <- train(y ~ .,   
# data = data.training,   
# method = "C5.0",  
# preProc = c("center", "scale"),   
# trControl = train.control,  
# metric = "ROC",  
# tuneGrid = c5Grid  
# )  
#stopImplicitCluster()  
#c5\_0.fit  
#saveRDS(c5\_0.fit, file = "c5\_0.fit.rds")  
  
c5\_0.fit <- readRDS(file = "c5\_0.fit.rds")  
get\_model\_results(c5\_0.fit, data.training, data.testing, TRUE, TRUE)

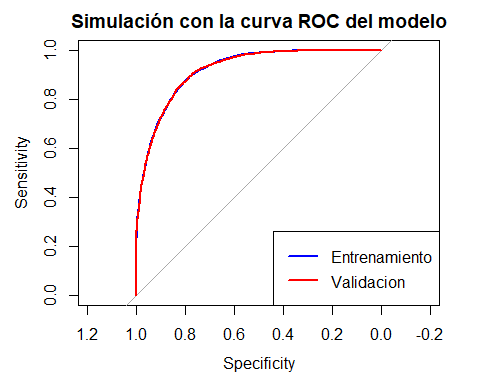
## [1] "#######################################################"  
## [1] "Matriz de confusión del modelo:"  
## [1] "#######################################################"  
## Cross-Validated (10 fold, repeated 5 times) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction no yes  
## no 40.8 7.4  
## yes 9.2 42.6  
##   
## Accuracy (average) : 0.8343  
##   
## [1] "#######################################################"  
## [1] "Matriz de confusión comparando las predicciones del modelo y los datos de test:"  
## [1] "#######################################################"  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 1901 328  
## yes 436 2009  
##   
## Accuracy : 0.8365   
## 95% CI : (0.8256, 0.847)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6731   
##   
## Mcnemar's Test P-Value : 0.0001083   
##   
## Sensitivity : 0.8134   
## Specificity : 0.8596   
## Pos Pred Value : 0.8528   
## Neg Pred Value : 0.8217   
## Prevalence : 0.5000   
## Detection Rate : 0.4067   
## Detection Prevalence : 0.4769   
## Balanced Accuracy : 0.8365   
##   
## 'Positive' Class : no   
##   
## [1] "#######################################################"  
## [1] "Importancia de las variables:"  
## C5.0 variable importance  
##   
## only 20 most important variables shown (out of 46)  
##   
## Overall  
## relationshipOwn.child 100.00  
## capital.gainHighCapitalGain 100.00  
## capital.loss 100.00  
## age 100.00  
## marital.statusMarried.AF.or.civ 100.00  
## marital.statusNever.married 99.22  
## hours.per.week 98.97  
## occupationFarming.fishing 98.87  
## educationHS.grad 95.93  
## occupationProf.specialty 93.85  
## relationshipOther.relative 89.89  
## occupationTransport.moving 87.51  
## educationDoctorate 87.01  
## occupationOther.service 86.40  
## UnploymentRatio 86.37  
## workclassSelf.emp.not.inc 86.03  
## occupationHandlers.cleaners 85.86  
## sexFemale 84.80  
## marital.statusSeparated 84.48  
## occupationPriv.house.serv 83.28  
## [1] "Gráfico ROC-parámetros:"



plot\_roc\_curve(c5\_0.fit, data.training, data.testing)

## [1] "Curva ROC:"

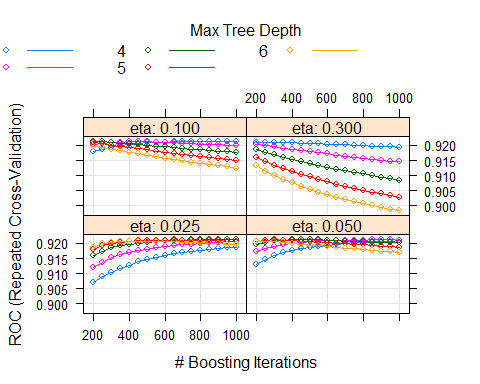
## [1] "Curva ROC para los datos de entrenamiento:"  
## Area under the curve: 0.9196  
## [1] "Curva ROC para los datos de validación:"  
## Area under the curve: 0.9188



## 8.6.1. Gradient Boosting

#set.seed(999)  
#nrounds <- 1000  
#xbgTreeGrid <- expand.grid( nrounds = seq(from = 200, to = nrounds, by = 50),  
# eta = c(0.025, 0.05, 0.1, 0.3),  
# max\_depth = c(2, 3, 4, 5, 6),  
# gamma = 0,  
# colsample\_bytree = 1,  
# min\_child\_weight = 1,  
# subsample = 1 )  
#clusterCPU <- registerDoParallel(cores=detectCores() - 1)  
#xbgTree.fit <- train(y ~ .,   
# data = data.training,   
# method = "xgbTree",  
# preProc = c("center", "scale"),   
# trControl = train.control,  
# metric = "ROC",  
# tuneGrid = xbgTreeGrid  
# )  
#stopImplicitCluster()  
#xbgTree.fit  
#saveRDS(xbgTree.fit, file = "xbgTree.rds")  
  
xbgTree.fit <- readRDS(file = "xbgTree.rds")  
get\_model\_results(xbgTree.fit, data.training, data.testing, TRUE, TRUE)

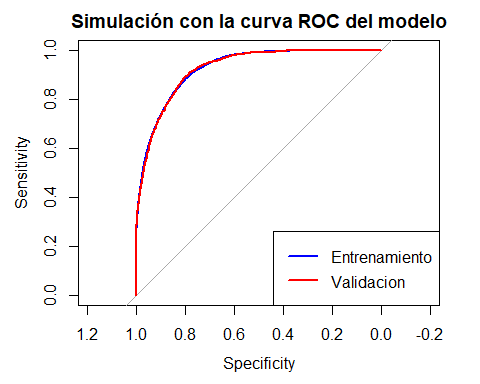
## [1] "#######################################################"  
## [1] "Matriz de confusión del modelo:"  
## [1] "#######################################################"  
## Cross-Validated (10 fold, repeated 5 times) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction no yes  
## no 40.3 6.5  
## yes 9.7 43.5  
##   
## Accuracy (average) : 0.8373  
##   
## [1] "#######################################################"  
## [1] "Matriz de confusión comparando las predicciones del modelo y los datos de test:"  
## [1] "#######################################################"  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 1883 271  
## yes 454 2066  
##   
## Accuracy : 0.8449   
## 95% CI : (0.8342, 0.8552)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6898   
##   
## Mcnemar's Test P-Value : 1.387e-11   
##   
## Sensitivity : 0.8057   
## Specificity : 0.8840   
## Pos Pred Value : 0.8742   
## Neg Pred Value : 0.8198   
## Prevalence : 0.5000   
## Detection Rate : 0.4029   
## Detection Prevalence : 0.4608   
## Balanced Accuracy : 0.8449   
##   
## 'Positive' Class : no   
##   
## [1] "#######################################################"  
## [1] "Importancia de las variables:"  
## xgbTree variable importance  
##   
## only 20 most important variables shown (out of 46)  
##   
## Overall  
## marital.statusMarried.AF.or.civ 100.0000  
## UnploymentRatio 32.0755  
## capital.gainHighCapitalGain 30.0258  
## age 25.5739  
## capital.loss 13.7928  
## hours.per.week 13.4567  
## educationElemtary.education 3.6959  
## occupationOther.service 3.3092  
## sexFemale 1.8820  
## occupationProf.specialty 1.8667  
## relationshipWife 1.3762  
## occupationExec.managerial.Machine.op.inspct 1.3718  
## occupationFarming.fishing 1.1698  
## relationshipNot.in.family 0.8516  
## educationBachelors 0.8165  
## occupationHandlers.cleaners 0.7573  
## workclassSelf.emp.not.inc 0.5571  
## educationProf.school 0.5118  
## relationshipUnmarried 0.5080  
## occupationTech.support 0.4931  
## [1] "Gráfico ROC-parámetros:"



plot\_roc\_curve(xbgTree.fit, data.training, data.testing)

## [1] "Curva ROC:"

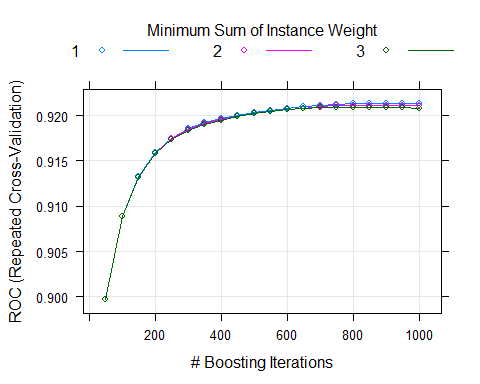
## [1] "Curva ROC para los datos de entrenamiento:"  
## Area under the curve: 0.9251  
## [1] "Curva ROC para los datos de validación:"  
## Area under the curve: 0.9246



## 8.6.2. Gradient Boosting

#set.seed(999)  
#nrounds <- 1000  
#xbgTreeGrid1 <- expand.grid( nrounds = seq(from = 50, to = nrounds, by = 50),  
# eta = xbgTree.fit$bestTune$eta,  
# max\_depth = ifelse( xbgTree.fit$bestTune$max\_depth == 2,  
# c(xbgTree.fit$bestTune$max\_depth:4),  
# xbgTree.fit$bestTune$max\_depth - 1:xbgTree.fit$bestTune$max\_depth + 1 ),  
# gamma = 0,  
# colsample\_bytree = 1,  
# min\_child\_weight = c(1, 2, 3),  
# subsample = 1 )  
#clusterCPU <- registerDoParallel(cores=detectCores() - 1)  
#xbgTree1.fit <- train(y ~ .,   
# data = data.training,   
# method = "xgbTree",  
# preProc = c("center", "scale"),   
## trControl = train.control,  
# metric = "ROC",  
# tuneGrid = xbgTreeGrid1  
# )  
#stopImplicitCluster()  
#xbgTree1.fit  
#saveRDS(xbgTree1.fit, file = "xbgTree1.rds")  
  
xbgTree1.fit <- readRDS(file = "xbgTree1.rds")  
get\_model\_results(xbgTree1.fit, data.training, data.testing, TRUE, TRUE)

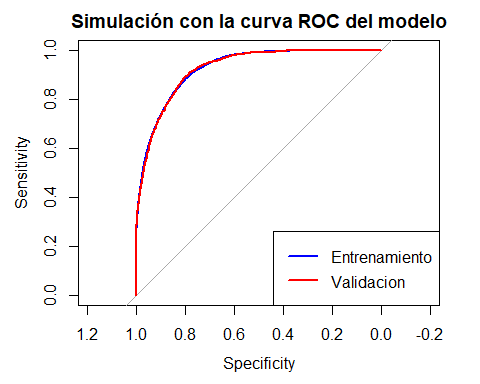
## [1] "#######################################################"  
## [1] "Matriz de confusión del modelo:"  
## [1] "#######################################################"  
## Cross-Validated (10 fold, repeated 5 times) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction no yes  
## no 40.3 6.5  
## yes 9.7 43.5  
##   
## Accuracy (average) : 0.8373  
##   
## [1] "#######################################################"  
## [1] "Matriz de confusión comparando las predicciones del modelo y los datos de test:"  
## [1] "#######################################################"  
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 1883 271  
## yes 454 2066  
##   
## Accuracy : 0.8449   
## 95% CI : (0.8342, 0.8552)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6898   
##   
## Mcnemar's Test P-Value : 1.387e-11   
##   
## Sensitivity : 0.8057   
## Specificity : 0.8840   
## Pos Pred Value : 0.8742   
## Neg Pred Value : 0.8198   
## Prevalence : 0.5000   
## Detection Rate : 0.4029   
## Detection Prevalence : 0.4608   
## Balanced Accuracy : 0.8449   
##   
## 'Positive' Class : no   
##   
## [1] "#######################################################"  
## [1] "Importancia de las variables:"  
## xgbTree variable importance  
##   
## only 20 most important variables shown (out of 46)  
##   
## Overall  
## marital.statusMarried.AF.or.civ 100.0000  
## UnploymentRatio 32.0755  
## capital.gainHighCapitalGain 30.0258  
## age 25.5739  
## capital.loss 13.7928  
## hours.per.week 13.4567  
## educationElemtary.education 3.6959  
## occupationOther.service 3.3092  
## sexFemale 1.8820  
## occupationProf.specialty 1.8667  
## relationshipWife 1.3762  
## occupationExec.managerial.Machine.op.inspct 1.3718  
## occupationFarming.fishing 1.1698  
## relationshipNot.in.family 0.8516  
## educationBachelors 0.8165  
## occupationHandlers.cleaners 0.7573  
## workclassSelf.emp.not.inc 0.5571  
## educationProf.school 0.5118  
## relationshipUnmarried 0.5080  
## occupationTech.support 0.4931  
## [1] "Gráfico ROC-parámetros:"



plot\_roc\_curve(xbgTree1.fit, data.training, data.testing)

## [1] "Curva ROC:"

## [1] "Curva ROC para los datos de entrenamiento:"  
## Area under the curve: 0.9251  
## [1] "Curva ROC para los datos de validación:"  
## Area under the curve: 0.9246



# 9. Comparativa de modelos

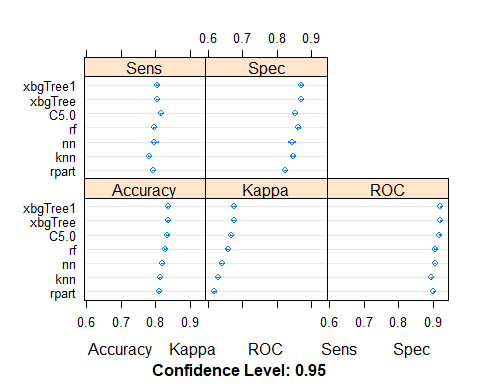
modelos <- list(xbgTree=xbgTree.fit, xbgTree1=xbgTree1.fit, rf=rf.fit, knn = knn.fit, nn = nn.fit, C5.0 = c5\_0.fit, rpart = rpart.fit)  
resultados <- resamples(modelos)  
resultados

##   
## Call:  
## resamples.default(x = modelos)  
##   
## Models: xbgTree, xbgTree1, rf, knn, nn, C5.0, rpart   
## Number of resamples: 50   
## Performance metrics: Accuracy, Kappa, ROC, Sens, Spec   
## Time estimates for: everything, final model fit

summary(resultados)

##   
## Call:  
## summary.resamples(object = resultados)  
##   
## Models: xbgTree, xbgTree1, rf, knn, nn, C5.0, rpart   
## Number of resamples: 50   
##   
## Accuracy   
## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
## xbgTree 0.8195931 0.8325532 0.8381487 0.8373444 0.8426471 0.8545455 0  
## xbgTree1 0.8195931 0.8325532 0.8381487 0.8373444 0.8426471 0.8545455 0  
## rf 0.8069519 0.8240642 0.8310160 0.8288937 0.8349606 0.8449198 0  
## knn 0.7919786 0.8085561 0.8133189 0.8140883 0.8223889 0.8326203 0  
## nn 0.7938972 0.8151070 0.8211230 0.8204850 0.8269588 0.8368111 0  
## C5.0 0.8110278 0.8290107 0.8342246 0.8342740 0.8399733 0.8508021 0  
## rpart 0.7858672 0.8049465 0.8098930 0.8096492 0.8163456 0.8282504 0  
##   
## Kappa   
## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
## xbgTree 0.6391863 0.6651153 0.6762969 0.6746888 0.6852941 0.7090909 0  
## xbgTree1 0.6391863 0.6651153 0.6762969 0.6746888 0.6852941 0.7090909 0  
## rf 0.6139037 0.6481283 0.6620321 0.6577874 0.6699150 0.6898396 0  
## knn 0.5839572 0.6171123 0.6266318 0.6281767 0.6447848 0.6652406 0  
## nn 0.5877944 0.6302139 0.6422460 0.6409698 0.6539177 0.6736227 0  
## C5.0 0.6220557 0.6580214 0.6684492 0.6685479 0.6799465 0.7016043 0  
## rpart 0.5717345 0.6098930 0.6197861 0.6192988 0.6326949 0.6564958 0  
##   
## ROC   
## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
## xbgTree 0.9020635 0.9173611 0.9220801 0.9213502 0.9258256 0.9320307 0  
## xbgTree1 0.9020635 0.9173611 0.9220801 0.9213502 0.9258256 0.9320307 0  
## rf 0.8861182 0.9022690 0.9059270 0.9061768 0.9106788 0.9189860 0  
## knn 0.8708520 0.8910040 0.8958838 0.8957806 0.9016034 0.9084141 0  
## nn 0.8855146 0.9012598 0.9055834 0.9050246 0.9104917 0.9158441 0  
## C5.0 0.8979814 0.9143009 0.9194008 0.9184718 0.9229723 0.9307936 0  
## rpart 0.8788717 0.8976665 0.9008310 0.9009872 0.9055142 0.9148852 0  
##   
## Sens   
## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
## xbgTree 0.7764706 0.7987690 0.8032086 0.8053268 0.8110788 0.8374332 0  
## xbgTree1 0.7764706 0.7987690 0.8032086 0.8053268 0.8110788 0.8374332 0  
## rf 0.7743316 0.7891356 0.7940077 0.7953358 0.8020861 0.8256684 0  
## knn 0.7475936 0.7743316 0.7806313 0.7827554 0.7881786 0.8181818 0  
## nn 0.7122995 0.7796791 0.7957219 0.7974126 0.8221925 0.8748663 0  
## C5.0 0.7925134 0.8085561 0.8142398 0.8166023 0.8232620 0.8566845 0  
## rpart 0.7518717 0.7871658 0.7945425 0.7946948 0.8074866 0.8179872 0  
##   
## Spec   
## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
## xbgTree 0.8374332 0.8652406 0.8705882 0.8693620 0.8759358 0.8877005 0  
## xbgTree1 0.8374332 0.8652406 0.8705882 0.8693620 0.8759358 0.8877005 0  
## rf 0.8235294 0.8576398 0.8636364 0.8624513 0.8695187 0.8855615 0  
## knn 0.8096257 0.8406417 0.8470588 0.8454215 0.8530909 0.8715203 0  
## nn 0.7347594 0.8248196 0.8422460 0.8435566 0.8681818 0.9262032 0  
## C5.0 0.8094218 0.8451459 0.8529412 0.8519453 0.8609626 0.8780749 0  
## rpart 0.7989305 0.8128342 0.8251337 0.8246047 0.8348945 0.8491979 0

dotplot(resultados)



Result <- function ( modelos ){  
 n\_modelos = length(modelos)  
 comparativa <- matrix(0, n\_modelos, 7)  
 pred <- NULL  
 data\_testing <- NULL  
 for (i in 1:n\_modelos){  
 if(modelos[[i]]$method == "custom") {  
 data\_testing <- data.testing.d  
 }  
 else {  
 data\_testing <- data.testing  
 }  
 pred[[i]] <- predict(modelos[i], data\_testing, type="prob")  
 comparativa[i,1] = modelos[[i]]$method  
 if (modelos[[i]]$method == "treebag"){  
 comparativa[i,2] = "-"  
 comparativa[i,3] = "-"  
 comparativa[i,4] = "-"  
 comparativa[i,5] = modelos[[i]]$results$Accuracy  
 comparativa[i,6] = modelos[[i]]$results$Kappa  
 }else{  
 comparativa[i,2] = modelos[[i]]$results[rownames(modelos[[i]]$bestTune), c("ROC")]  
 comparativa[i,3] = modelos[[i]]$results[rownames(modelos[[i]]$bestTune), c("Sens")]  
 comparativa[i,4] = modelos[[i]]$results[rownames(modelos[[i]]$bestTune), c("Spec")]  
 comparativa[i,5] = modelos[[i]]$results[rownames(modelos[[i]]$bestTune), c("Accuracy")]  
 comparativa[i,6] = modelos[[i]]$results[rownames(modelos[[i]]$bestTune), c("Kappa")]  
 }  
 comparativa[i,7] = auc(roc(data\_testing$y,pred[[i]][[1]][,"yes"]))  
 }  
 colnames(comparativa) <- c("Modelo", "ROC", "Sens", "Spec", "Accuracy", "Kappa", "ROC Validación")  
 return(comparativa)  
}  
  
Result(modelos)

## Modelo ROC Sens Spec   
## [1,] "xgbTree" "0.921350159094819" "0.805326752854149" "0.869361998877807"  
## [2,] "xgbTree" "0.921350159094819" "0.805326752854149" "0.869361998877807"  
## [3,] "parRF" "0.906176818271324" "0.795335844908335" "0.862451327737636"  
## [4,] "knn" "0.895780551811663" "0.782755442063919" "0.84542154381706"   
## [5,] "mlp" "0.905024565162643" "0.797412566272372" "0.843556642123464"  
## [6,] "C5.0" "0.918471809437904" "0.816602319962441" "0.851945310263486"  
## [7,] "rpart" "0.90098722814039" "0.794694752029681" "0.824604701760011"  
## Accuracy Kappa ROC Validación   
## [1,] "0.837344419574419" "0.674688807446843" "0.924571309087187"  
## [2,] "0.837344419574419" "0.674688807446843" "0.924571309087187"  
## [3,] "0.828893722031139" "0.65778736196339" "0.935363445925521"  
## [4,] "0.814088286124939" "0.628176689792341" "0.905477711624627"  
## [5,] "0.820485020929169" "0.640969777892698" "0.912634446255279"  
## [6,] "0.834274013540953" "0.668547895070111" "0.918846031241206"  
## [7,] "0.809649240269301" "0.619298845674651" "0.91546659943324"

# 10. Elección del modelo e interpretación de los resultados

Aunque no se refleja en los datos de entrenamiento, claramente el mejor resultado usando la métrica ROC se consigue con el modelo random forest, y esto queda reflejado en casi un 87% de exactitud del modelo y una métrica ROC de 0.93. Random forest combina modelos tipo árbol como el CART mediante métodos de ensemble. Para medir la importancia de las variables, se evalúa cómo cada característica reduce la impureza entre los árboles que combina random forest, tal y como se explica en [6]. Por tanto, dan un criterio válido para el objetivo de detectar las métricas más importantes, aunque no de una forma tan nítida como los árboles CART, C50 o CHAID. Aunque la interpretación correcta y profunda de los resultados depende en muchos casos de ojos expertos en la materia, destacamos las métricas más importantes reflejadas por la estructura predictiva del modelo, dando una interpretación, posiblemente subjetiva y a contrastar por juicios más expertos en la materia:

* Vemos en las primeras posiciones, no sólo del algoritmo seleccionado, sino de casi todos los algoritmos la categoría de estado civil casado con o sin otra familia junto con la condición de cónjuge como uno de los factores más importantes. El patrón subyacente parece indicar un perfil de marido-esposa integrado en una unidad familiar. Por tanto, queda reflejado un primer indicador de los buscasdos. A nivel interpretativo, una posible hipóstesis que explica este fenómeno es que la familia es incentivo de gasto y, por lo tanto, de motor para escalar hacia nuevos niveles salariales. Aunque esta hipótesis debería ser más desarrollada, queda fuera de los alcances del presente trabajo y sólo se insinúa a modo de conjetura.
* A pesar de que no se ha representado al nivel de granularidad adecuado y que, por lo tanto, no recoge toda la variabilidad posible, el nivel de desempleo por estratos educativos se muestra como otra métrica muy importante. Posiblemente, esto se deba al principio de oferta demanda aplicado al ámbito laboral
* Otra métrica que aparece de forma muy clara es la tenencia de capital (capital.gain, capital.loss). De hecho, en el transcurso de la búsqueda de patrones, se detectó un punto de corte significativo en la variable de ganancia de capital, demostrando que este factor es, por sí solo, de vital importancia, pues era capaz de recudir la impureza de manera muy notable. Quizás la interpretación de este factor es más evidente, indicando una actividad de capitalización de bienes (como un alquiler de un piso en propiedad, un almacén, etc).
* Igualmente se ven otros factores relevantes, aunque a un nivel de relevancia menor respecto a los primeros mencionados como son el nivel formativo, la clase trabajadora-ocupación y el sexo, pudiendo ésta tener como causa la desigualdad de género a nivel salarial.

# 11. Conclusiones finales

De esta forma, mediante el tratamiento adecuado de la base de datos, representando los patrones descubiertos en la exploración de datos y la descrpción de la estructura predictiva se ha conseguido el doble objetivo, obtiendo un estimador equilibrado y con una buena capacidad de predicción. Además, describiendo la estrucutura predictiva del modelo seleccionado, se han detectado las métricas más importantes. Éstas pueden ser usadas como entradas pasa otros estudios, incluyendo una nueva iteración del descubrimiento de conocimiento para la base de datos del presente estudio. Como nota final, notar que se ha eliminado la variable fnlwgt por no tener una interpretabilidad clara y por no estar claro que pueda obtenerse este dato para realizarse predicciones. Si se quiere únicamente mejorar la precisión, pueden probarse modelos introduciendo dicha variable.

# 12. Referencias bibliográficas

[1] <https://archive.ics.uci.edu/ml/datasets/Census+Income>

[2] <https://www.bls.gov/cps/cpsatabs.htm>

[3] G. V. Kass. 1980. An exploratory Technique for Investigating Large Quantities Of categorical Data. Universidad de Witwatersrand, Sur África. Journal of the Royal Statistical Society. Series C (Applided statistics), Vol. 29, No. 2, pp. 119-127.

[4] Breiman et al., 1984, Clasification And Regression Trees. U.S. Chapman & Hall

[5] <https://christophm.github.io/interpretable-ml-book/interaction.html>

[6] Breiman, 2001, Random Forest. Holanda Springer Link, sitio web <https://link.springer.com/article/10.1023/A:1010933404324>.