Examen clasificacion

Alvaro Ferro Perez

05/02/2019

EXAMEN de Técnicas de Clasificación 2018/2019

Exploración y preparación de los datos

Para la resolución del examen de Técnicas de Clasificación, se va a proceder, en primer lugar, a realizar la carga de librerías que se utilizarán a lo largo del desarrollo del mismo.

Resumen ejecutivo El objetivo del siguiente informe será realizar una comparativa entre trés técnicas para ver cual es mejor para realizar una predicción o clasificación de una nueva observación.

Usaremos los modelos de regresión logística y árboles de clasificación para llevar a cabo nuestro análisis y comparación.

A tener en cuenta es que este documento contiene todo el código que se ha usado y algunos comentarios. Se deberá acompañar este documento del informe de negocio para obtener el 100% de información.

INTRODUCCION

La encuesta de Presupuestos familiares (EPF) es realizada cada año por el Instituto nacional de estadística, siendo su objetivo conocer las comparativas con el fin de conseguir el casto en consumo de los hogares españoles, asi com la distribución del mismo entre las diversas parcelas consumistas, de esa manera reemplaza a la Encuesta continua de Presupuestos Familiares(ECPF) que estuvo en vigor desde el año 1997 al 2005 incorporando diversas mejora meteorologicas, tales como el cambio de periodicidad (de trimestral a anual), asi como el aumento del tamaño de la muestra.

La EPF otorga la informacion imprescindible de las estimaciones sobre el gasto en consumo de los hogares de la contabilidad Nacional y para actualizar las ponderaciones del indice deprecios al Consumo(IPC).

Los gastos de consumo que se registran en la EPF se refieren tanto al flujo monetario que destina el hogar, en este caso, single, al pago de determinados bienes y servicios de consumo final asi como al valor de determinados consumos no monetarios efectuados por los hogares. Estos ultimos son por ejemplo nuestra variable 'REGTEN'la cual tomarÃ¡ valor 0 en caso de alquiler imputado/pago de hipoteca.

Las variables objeto de estudio son las siguientes:

Table 1: Tabla de las variables objeto de estudio

VARIABLES	DEFINICION
cat2	Variable de clasificacion de hogares segun su gasto en consumo de vacuno anual
cat3	Variable de clasificacion de hogares segun su gasto en consumo de vacuno anual
TAMAMU	Tamaño de los municipios
DENSIDAD	Densidad de la poblacion
EDAD	Edad, expresada en años
SEXO	Sexo de la muestra
ESTUD	Nivel de estudios completados
LAB	Situacion laboral
REGTEN	Regimen de tenencia de la vivienda
SUPERF	Superficie de la vivienda en metros cuadrados
IMPEXAC	Importe exacto de los ingresos mensuales netos totales del hogar en cientos

Todos estos datos fueron extraidos de una muestra de 4220 hogares distribuidos por todo el territorio nacional,

para comenzar, realizaremos un análisis exploratorio con la finalidad de identificar las variables explicativas, la variable dependiente, modificaciones necesarias para la base de datos...

Análisis exploratorio

Seguidamente, se va a realizar la carga de los datos y se hará un escueto análisis exploratorio y una limpieza pertinente para trabajar correctamente la misma.

```
set.seed(1234)
str(datos_train)
```

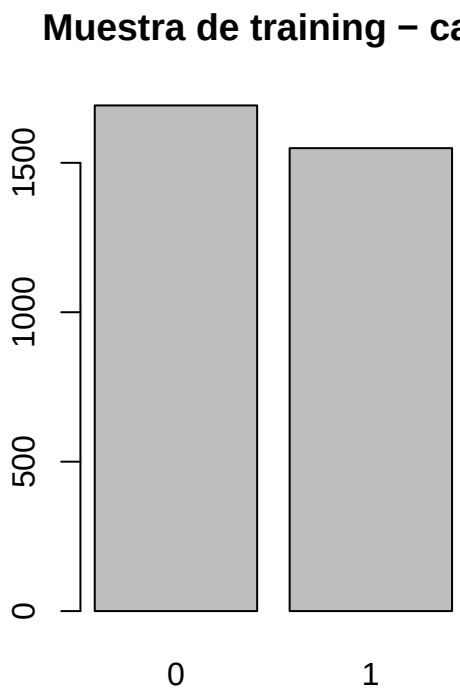
```
## Classes 'tbl_df', 'tbl' and 'data.frame':    3241 obs. of  11 variables:
##  $ TAMAMU  : Factor w/ 2 levels "0","1": 2 2 1 2 2 2 2 2 1 1 ...
##  $ DENSIDAD: Factor w/ 3 levels "1","2","3": 3 1 2 1 1 2 1 2 3 3 ...
##  $ EDAD    : num  85 81 39 84 65 85 39 84 58 25 ...
##  $ SEXO    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ ESTUD   : Factor w/ 4 levels "1","2","3","4": 1 2 4 2 4 1 3 1 4 3 ...
##  $ LAB     : Factor w/ 4 levels "1","2","3","4": 4 4 1 4 4 4 4 4 1 2 ...
##  $ REGTEN  : Factor w/ 2 levels "0","1": 1 2 1 2 2 2 1 2 2 2 ...
##  $ SUPERF  : num  90 90 72 70 85 80 50 98 80 88 ...
##  $ IMPEXAC : num  8.18 10.61 26.29 8 16.87 ...
##  $ cat2    : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 1 1 2 1 ...
##  $ cat3    : Factor w/ 3 levels "1","2","3": 1 2 2 2 3 2 1 1 2 2 ...
```

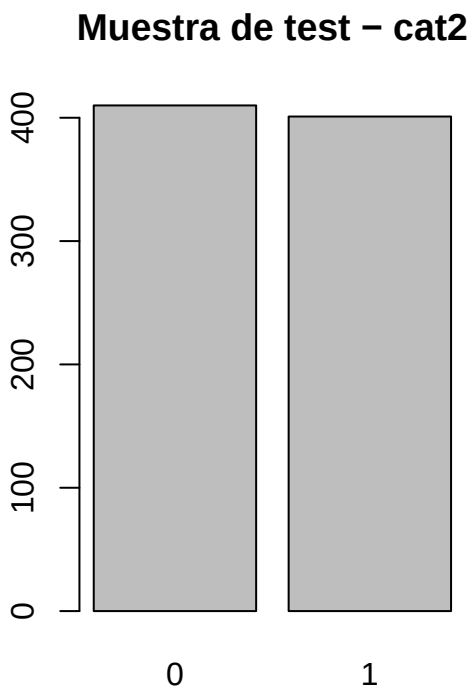
```
summary(datos)
```

```
##  TAMAMU   DENSIDAD      EDAD       SEXO     ESTUD    LAB      REGTEN
##  0: 887   1:2126   Min.   :18.00   0:2365   1:1130   1:1334   0:1312
##  1:3165   2: 875   1st Qu.:50.00   1:1687   2:1085   2: 144   1:2740
##           3:1051   Median :63.00            3: 691   3: 341
##                    Mean   :61.43            4:1146   4:2233
##                    3rd Qu.:75.00
##                    Max.   :85.00
##      SUPERF          IMPEXAC       cat2     cat3
##  Min.   : 35.00   Min.   :  0.00   0:2102   1:1390
##  1st Qu.: 67.00   1st Qu.:  7.50   1:1950   2:1416
##  Median : 85.00   Median : 10.50            3:1246
##  Mean   : 90.95   Mean   : 11.94
##  3rd Qu.:100.00   3rd Qu.: 15.00
##  Max.   :300.00   Max.   :152.07
```

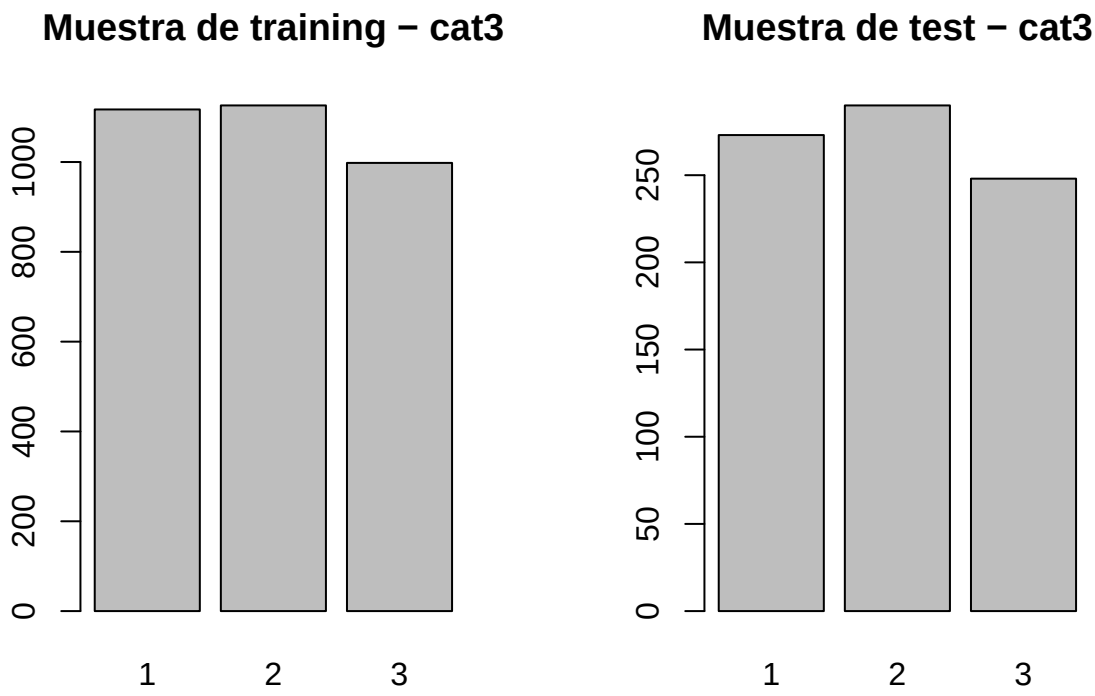
Una vez los datos han sido limpiados y preparados, se representan las variables a predecir o clasificar tanto para el train como para el test con el objetivo de ver si la muestra está balanceada.

```
set.seed(1234)
par(mfrow = c(1,2))
plot(as.factor(datos_train$cat2), main = "Muestra de training - cat2")
plot(as.factor(datos_test$cat2), main = "Muestra de test - cat2")
```






```
par(mfrow = c(1,2))
plot(as.factor(datos_train$cat3), main = "Muestra de training - cat3")
plot(as.factor(datos_test$cat3), main = "Muestra de test - cat3")
```



Estos graficos sirven para observar como están de balanceadas nuestras muestras. En este caso, lo hacemos tanto para la variable cat 2 como cat3 que como podemos observar ambas estan balanceadas.

En este punto, tenemos la base de datos lista para realizar todos los analisis.

```
set.seed(1234)
datos_convertidos$Ingresos <- datos_convertidos$Ingresos * 100
datos_convertidos <- datos_convertidos[, -10]
```

```
set.seed(1234)
str(datos_convertidos)
```

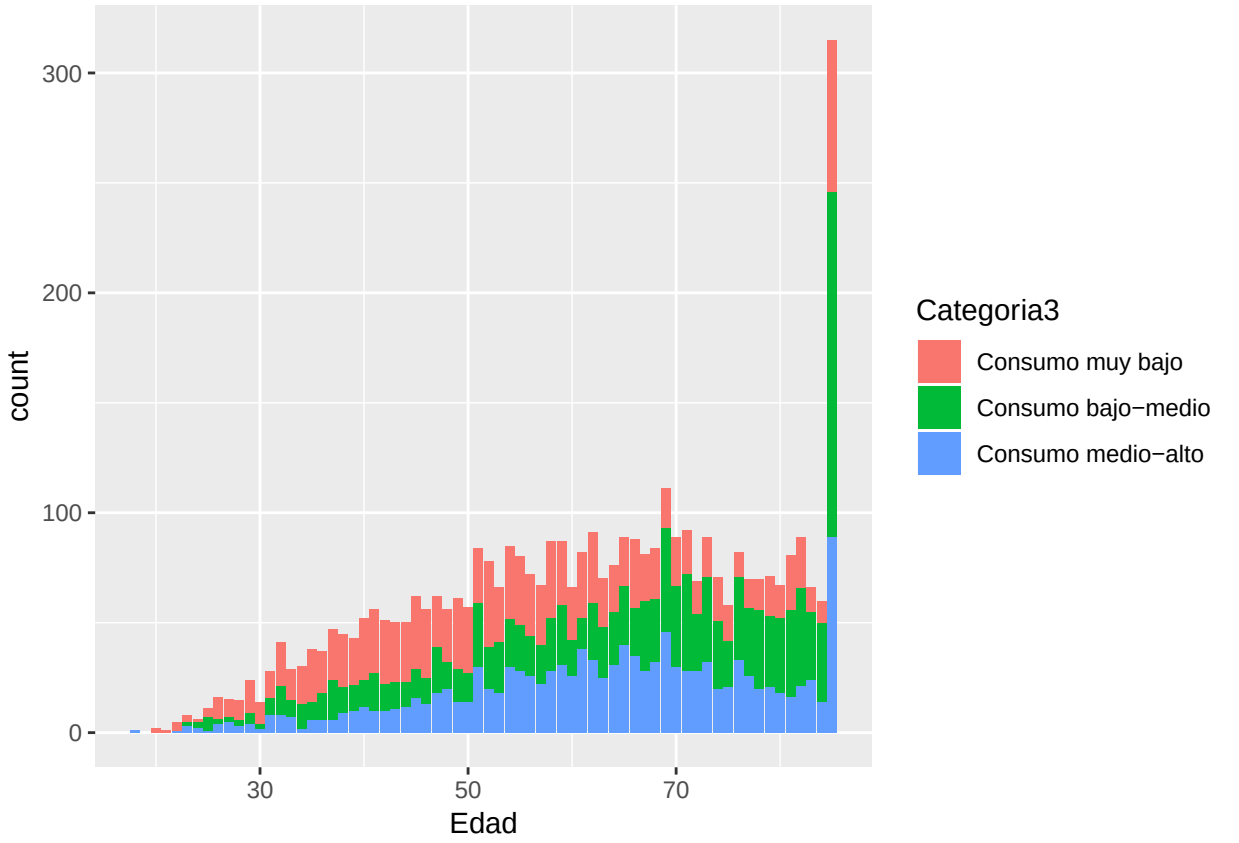
```
## Classes 'tbl_df', 'tbl' and 'data.frame':    4052 obs. of  10 variables:
##  $ TamanoMunicipio : Factor w/ 2 levels "Menos de 10.000 habs",..: 1 2 2 1 2 2 2 1 2 2 ...
##  $ DensidadZona    : Factor w/ 3 levels "Densamente poblada",..: 2 1 2 3 2 2 1 3 1 1 ...
##  $ Edad            : num  74 51 54 42 54 58 45 72 41 85 ...
##  $ Sexo            : Factor w/ 2 levels "Mujer","Hombre": 1 2 1 2 1 1 1 1 2 1 ...
##  $ Estudios        : Factor w/ 4 levels "Inferior ESO",..: 2 2 1 1 3 2 2 2 4 1 ...
##  $ SituacionLaboral: Factor w/ 4 levels "Ocupado JC","Ocupado JP",..: 4 3 3 1 3 3 3 4 1 4 ...
##  $ Vivienda        : Factor w/ 2 levels "Alquiler/Hipoteca",..: 2 2 2 2 2 2 1 1 1 1 ...
##  $ Superficie      : num  78 78 91 150 90 90 60 75 95 50 ...
##  $ Ingresos        : num  657 0 426 0 0 426 550 743 350 742 ...
##  $ Categoria3      : Factor w/ 3 levels "Consumo muy bajo",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
datos_adisc <- datos_convertidos
```

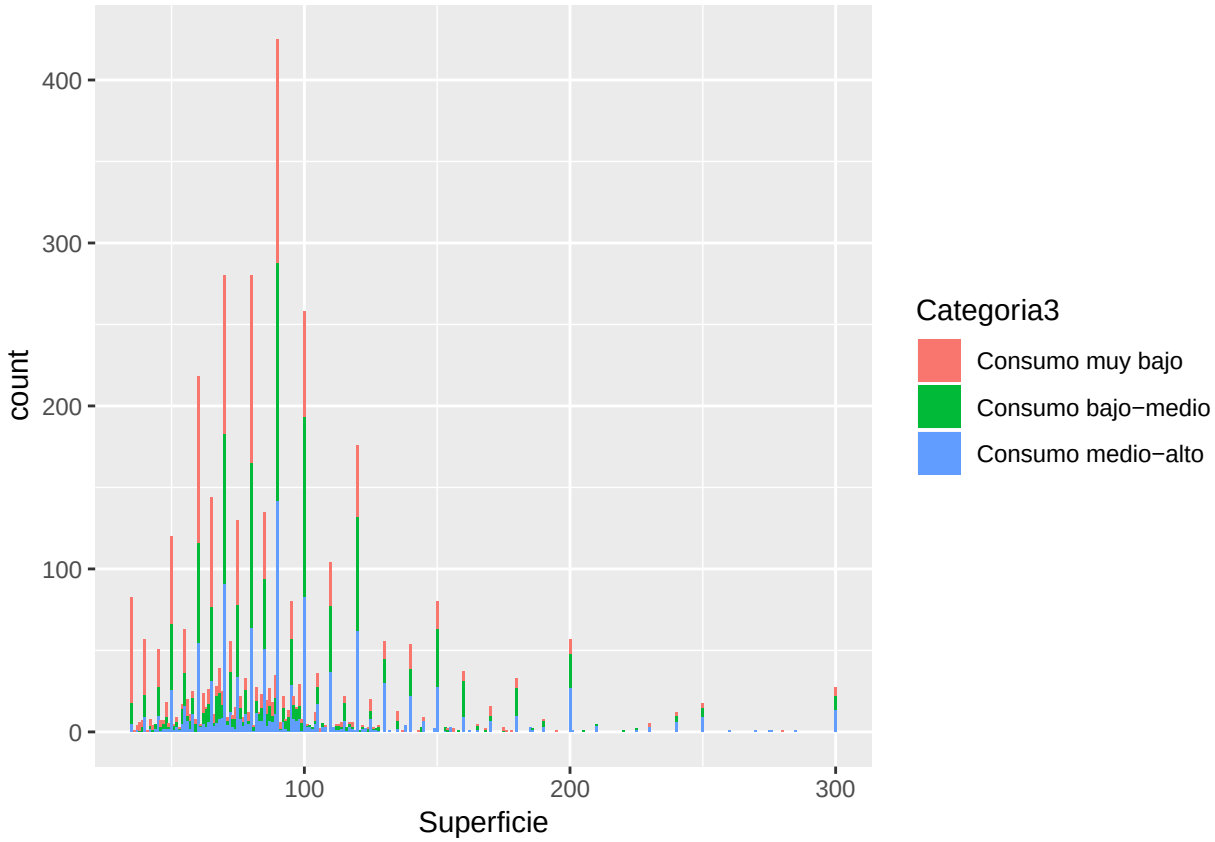
Para poder comprobar la normalidad de las variables, estas van a ser representadas. Las variables de tipo factor también van a ser representadas con gráficos de barras de colores. Se comprobará si existen grandes diferencias entre los consumos de vacuno (cat3) para las diferentes categorías de las variables.

- (num) En referencia a la Edad, la distribución está claramente desviada hacia la derecha. No parece que haya un reparto heterogéneo según categoría.
- (num) En referencia a la Superficie, la distribución parece normal, pero tiene algún atípico en la derecha.
- (num) En referencia a los Ingresos, la distribución parece normal.
- (factor) En referencia al Tamaño del municipio, no parece que hayan grandes diferencias entre los municipios pequeños y grandes.
- (factor) En referencia a la Densidad de la zona, tampoco parece que haya grandes diferencias.
- (factor) En referencia al Sexo, tampoco hay gran diferencia.
- (factor) En referencia al Nivel de estudios, sí que se aprecia diferencia entre los que más estudios tienen y los que menos.
- (factor) En referencia a la Situación laboral, sí que se aprecia diferencia entre los inactivos y ocupados a tiempo completo por un lado, y los otros por otro lado.
- (factor) En referencia al régimen de Vivienda, sí que hay diferencia.

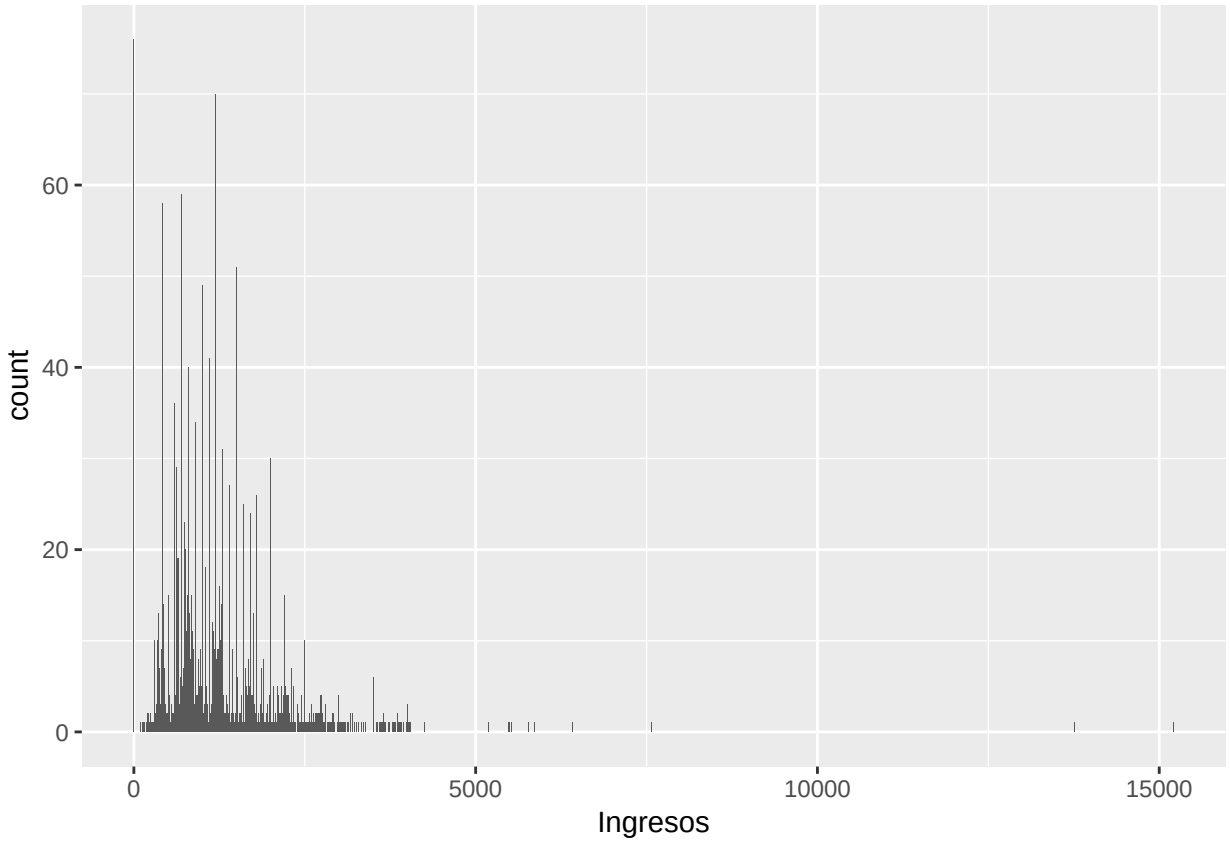
```
library(ggplot2)
ggplot(data = datos_adisc, mapping = aes(x = Edad)) +
  geom_bar(mapping = aes(fill = Categoria3))
```



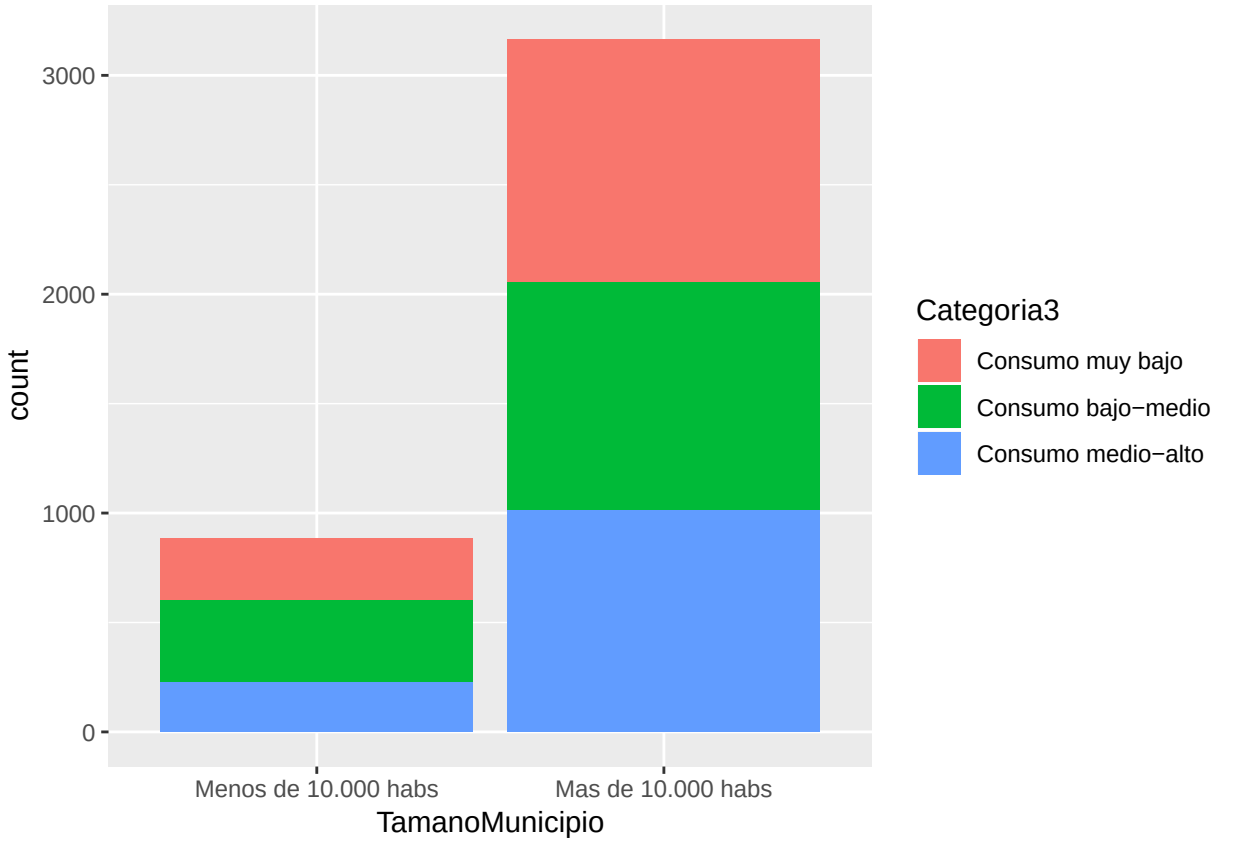

```
ggplot(data = datos_adisc, mapping = aes(x = Superficie)) +
  geom_bar(mapping = aes(fill = Categoria3))
```



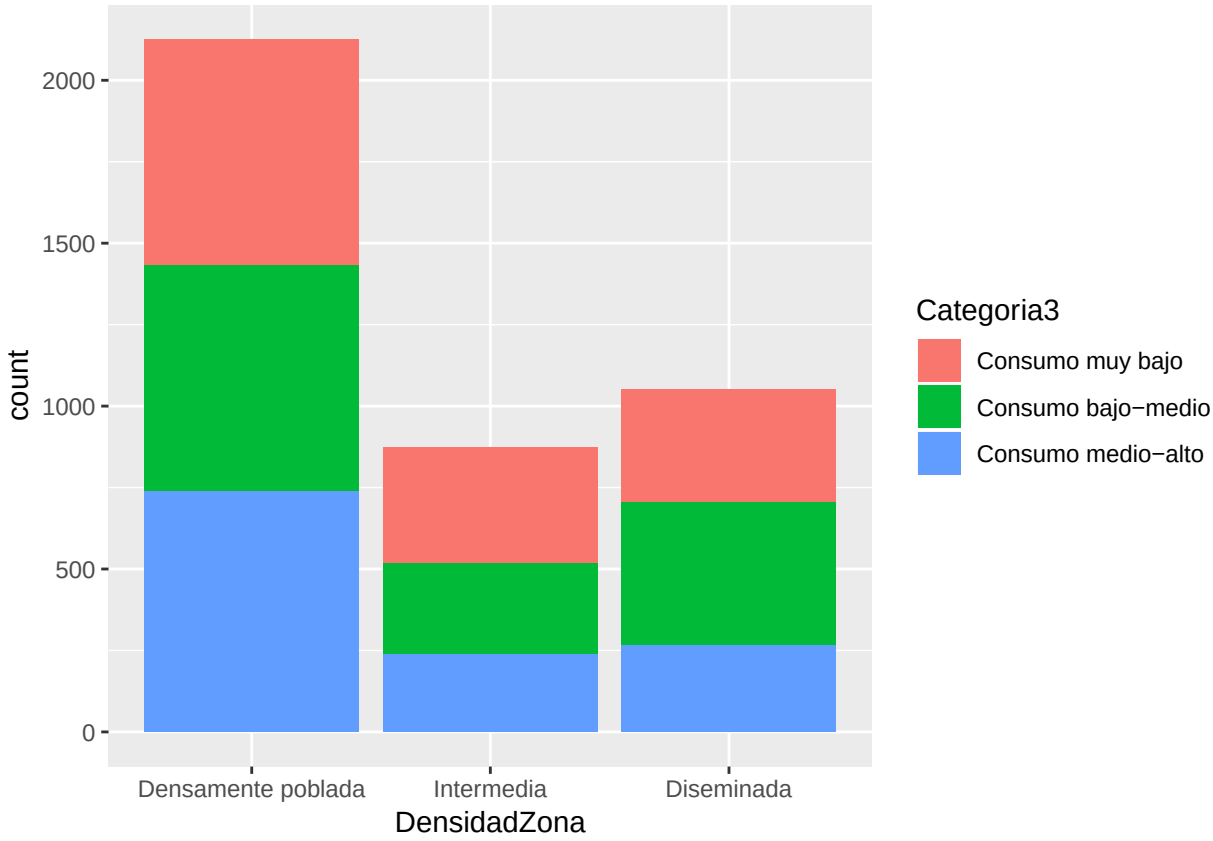

```
ggplot(data = datos_adisc, mapping = aes(x = Ingresos)) +
  geom_bar()
```



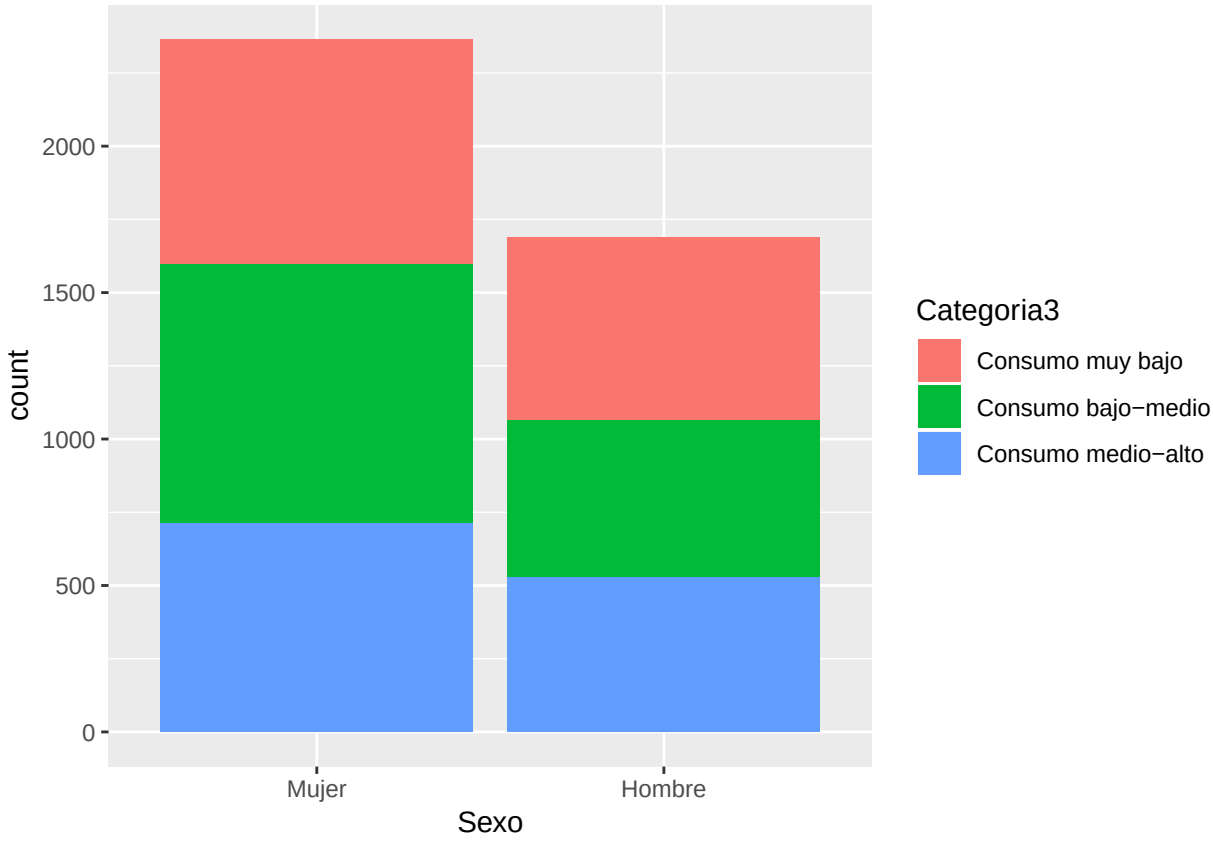

```
ggplot(data = datos_adisc, mapping = aes(x = TamanoMunicipio)) +
  geom_bar(mapping = aes(fill = Categoria3))
```



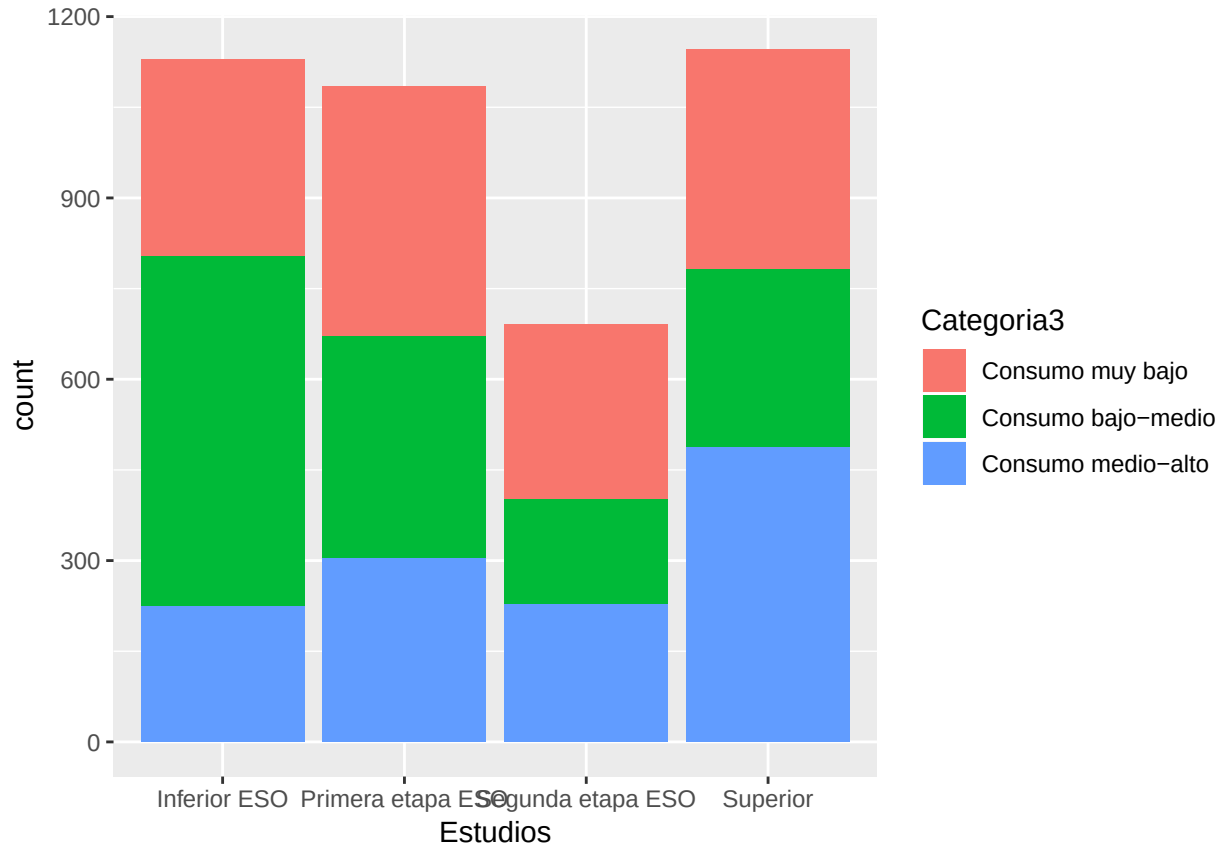

```
ggplot(data = datos_adisc, mapping = aes(x = DensidadZona)) +
  geom_bar(mapping = aes(fill = Categoria3))
```



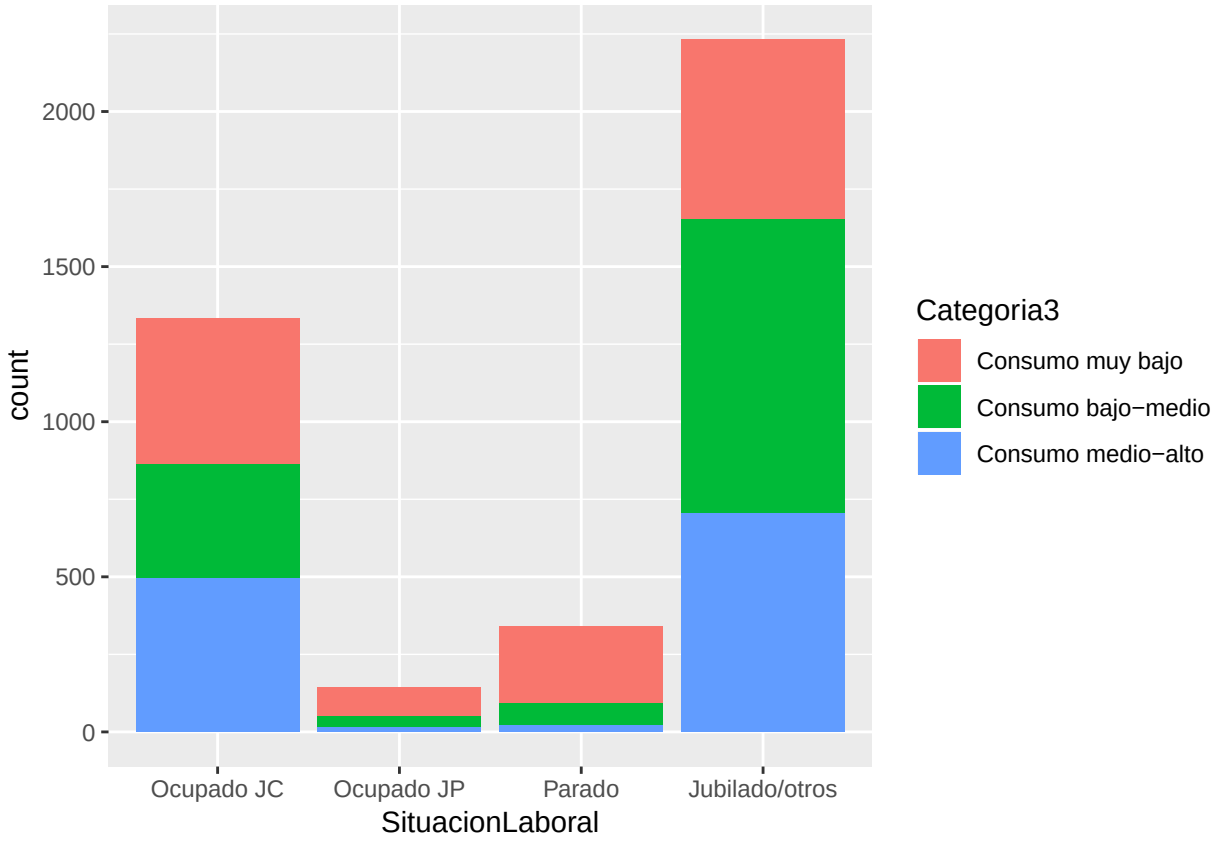

```
ggplot(data = datos_adisc, mapping = aes(x = Sexo)) +
  geom_bar(mapping = aes(fill = Categoria3))
```



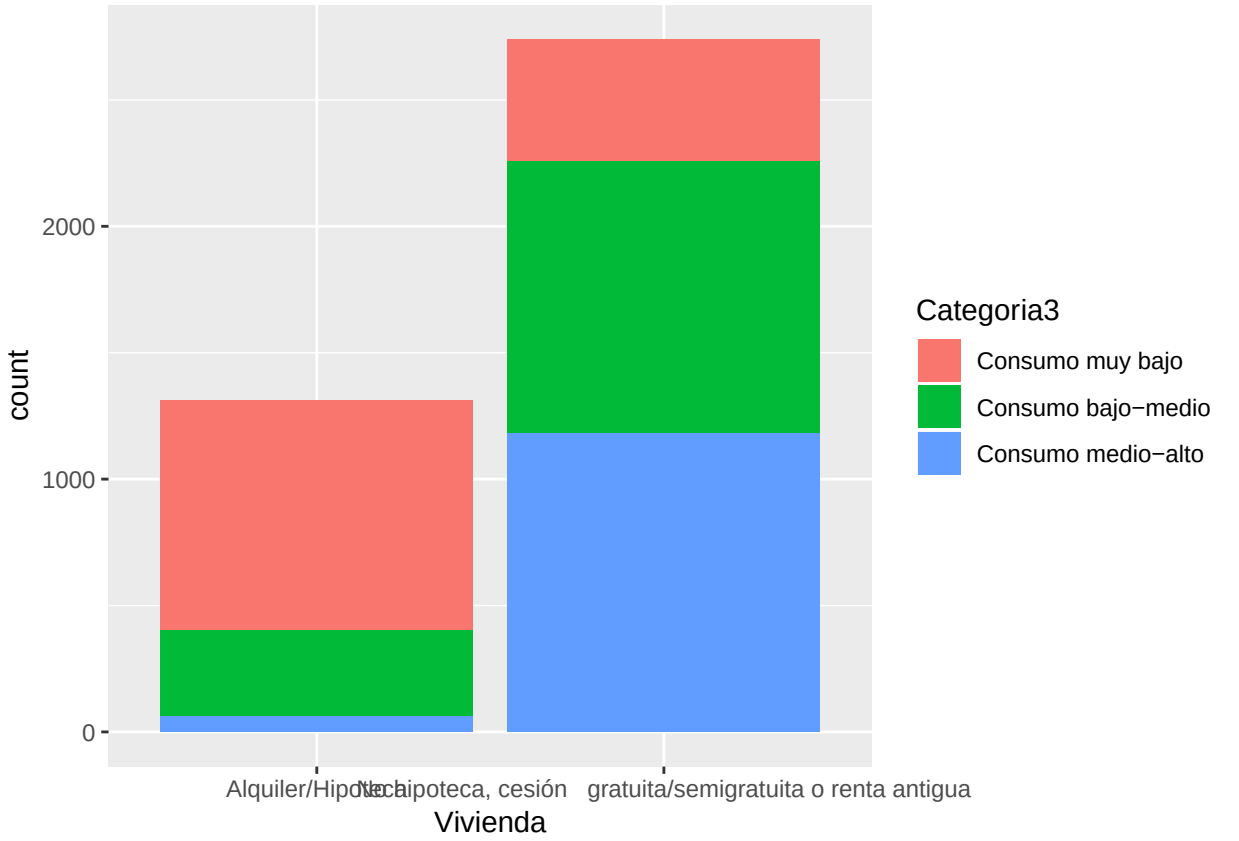

```
ggplot(data = datos_adisc, mapping = aes(x = Estudios)) +
  geom_bar(mapping = aes(fill = Categoria3))
```




```
ggplot(data = datos_adisc, mapping = aes(x = SituacionLaboral)) +
  geom_bar(mapping = aes(fill = Categoria3))
```




```
ggplot(data = datos_adisc, mapping = aes(x = Vivienda)) +
  geom_bar(mapping = aes(fill = Categoria3))
```



Aunque a priori parece que está demostrada la no normalidad de las variables, se va a realizar el test de Saphiro para corroborarlo. Evidentemente se confirma la no normalidad de todas las variables numéricas. H0: Es normal, H1: No es normal. Por tanto buscaremos rechazar la hipótesis nula, queremos que nuestros datos no sean normales.

```
set.seed(1234)
data_disc_num <- datos_adisc[, c(3, 8:10)]

cons_muybajo <- data_disc_num %>%
  filter(Categoria3 == "Consumo muy bajo")
cons_bajomedio <- data_disc_num %>%
  filter(Categoria3 == "Consumo bajo-medio")
cons_medioalto <- data_disc_num %>%
  filter(Categoria3 == "Consumo medio-alto")

cons_muybajo <- t(cons_muybajo[,-4])
mshapiro.test(cons_muybajo)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  Z
## W = 0.85754, p-value < 2.2e-16
```

```
cons_bajomedio <- t(cons_bajomedio[,-4])
mshapiro.test(cons_bajomedio)
```

```
##
```

```
##  Shapiro-Wilk normality test
##
## data:  Z
## W = 0.90692, p-value < 2.2e-16
```

```
cons_medioalto <- t(cons_medioalto[,-4])
mshapiro.test(cons_medioalto)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  Z
## W = 0.73132, p-value < 2.2e-16
```

Además, con el fin de corregir la normalidad, se van a escalar las variables.

```
set.seed(1234)
data_disc_num$Edad <- as.integer(data_disc_num$Edad)
data_disc_num$Superficie <- as.integer(data_disc_num$Superficie)
data_disc_num$Ingresos <- as.integer(data_disc_num$Ingresos)
str(data_disc_num)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    4052 obs. of  4 variables:
##  $ Edad      : int  74 51 54 42 54 58 45 72 41 85 ...
##  $ Superficie: int  78 78 91 150 90 90 60 75 95 50 ...
##  $ Ingresos  : int  657 0 426 0 0 426 550 743 350 742 ...
##  $ Categoria3: Factor w/ 3 levels "Consumo muy bajo",..: 1 1 1 1 1 1 1 1 1 1 ...
```

En estadística, la prueba de Bartlett se utiliza para probar si k muestras provienen de poblaciones con la misma varianza. A las varianzas iguales a través de las muestras se llama homocedasticidad u homogeneidad de varianzas, es sensible a las desviaciones de la normalidad.

```
set.seed(1234)
bartlett.test(data_disc_num)
```

```
## Warning in FUN(X[[i]], ...): Calling var(x) on a factor x is deprecated and will become an error.
##   Use something like 'all(duplicated(x)[-1L])' to test for a constant vector.
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  data_disc_num
## Bartlett's K-squared = 88302, df = 3, p-value < 2.2e-16
```

REGRESIÓN LOGÍSTICA Por tanto, lo que se buscará en el siguiente análisis será realizar un modelo de regresión logística con dichos datos, utilizando como variable a explicar "cat2 consumo medio-alto, medio-bajo", que es de tipo categórico.

A través de estos modelos podemos utilizar para los mismos variables cualitativas o categóricas como es el caso de nuestra variable, fragmentaremos el total de las informaciones en 80% que será la parte de entrenamiento, y un 20% lo de dedicaremos a la parte de testeo

Recordemos que un modelo de clasificación es aquel capaz de predecir a qué clase va a pertenecer una nueva instancia, basándose en lo aprendido en instancias anteriores, por tanto, realizaremos una matriz de confusión para observar si se han clasificado bien las variables o no si existen falsos positivos o falsos negativos. Y finalmente para evaluar este modelo podríamos simplemente calcular su precisión ("accuracy"), como la proporción entre las predicciones correctas que ha hecho el modelo y el total de predicciones.

```
set.seed(1234)
```

```r
datos_train_glm <- datos_train[,-11]

regresion <- glm(cat2 ~ ., family = "binomial", data = datos_train_glm)
```

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

```r
summary(regresion)
```

```
##
## Call:
## glm(formula = cat2 ~ ., family = "binomial", data = datos_train_glm)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9788  -0.4252  -0.0215   0.5605   3.4707
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -7.7302856  0.5245833 -14.736  < 2e-16 ***
## TAMAMU1     -0.2717277  0.1780683  -1.526  0.12702
## DENSIDAD2   -0.3879054  0.1423561  -2.725  0.00643 **
## DENSIDAD3   -0.3713914  0.1824911  -2.035  0.04184 *
## EDAD         0.0007914  0.0061227   0.129  0.89716
## SEXO1       -0.2905455  0.1162464  -2.499  0.01244 *
## ESTUD2       0.2342276  0.1351591   1.733  0.08310 .
## ESTUD3       0.1998424  0.1994679   1.002  0.31640
## ESTUD4      -0.1027133  0.2161022  -0.475  0.63457
## LAB2        -1.7113650  0.3510505  -4.875 1.09e-06 ***
## LAB3        -2.0728934  0.2873811  -7.213 5.47e-13 ***
## LAB4        -1.2165491  0.2316967  -5.251 1.52e-07 ***
## REGTEN1      6.8067800  0.2977157  22.863  < 2e-16 ***
## SUPERF       0.0008543  0.0014404   0.593  0.55309
## IMPEXAC      0.3271857  0.0174388  18.762  < 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: 4486.7  on 3240  degrees of freedom
## Residual deviance: 2259.5  on 3226  degrees of freedom
## AIC: 2289.5
##
## Number of Fisher Scoring iterations: 7
```

```r
stepAIC(regresion, direction = c("both"))
```

```
## Start:  AIC=2289.45
## cat2 ~ TAMAMU + DENSIDAD + EDAD + SEXO + ESTUD + LAB + REGTEN +
##     SUPERF + IMPEXAC
```

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

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

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

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

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

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

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
##            Df Deviance    AIC
## - EDAD      1   2259.5 2287.5
## - SUPERF    1   2259.8 2287.8
## - ESTUD     3   2265.1 2289.1
## <none>          2259.5 2289.5
## - TAMAMU    1   2261.8 2289.8
## - SEXO      1   2265.7 2293.7
## - DENSIDAD  2   2268.0 2294.0
## - LAB       3   2328.8 2352.8
## - IMPEXAC   1   2898.4 2926.4
## - REGTEN    1   3746.9 3774.9
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
##
## Step:  AIC=2287.47
## cat2 ~ TAMAMU + DENSIDAD + SEXO + ESTUD + LAB + REGTEN + SUPERF +
##     IMPEXAC
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

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

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

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

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

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

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

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
##            Df Deviance    AIC
## - SUPERF    1   2259.8 2285.8
## - ESTUD     3   2265.1 2287.1
## <none>          2259.5 2287.5
## - TAMAMU    1   2261.8 2287.8
## + EDAD      1   2259.5 2289.5
## - SEXO      1   2266.1 2292.1
## - DENSIDAD  2   2268.1 2292.1
## - LAB       3   2331.4 2353.4
## - IMPEXAC   1   2914.9 2940.9
## - REGTEN    1   3808.4 3834.4
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=2285.84
## cat2 ~ TAMAMU + DENSIDAD + SEXO + ESTUD + LAB + REGTEN + IMPEXAC
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
##            Df Deviance    AIC
## - ESTUD     3   2265.5 2285.5
## <none>          2259.8 2285.8
## - TAMAMU    1   2262.3 2286.3
## + SUPERF    1   2259.5 2287.5
## + EDAD      1   2259.8 2287.8
## - DENSIDAD  2   2268.1 2290.1
## - SEXO      1   2266.4 2290.4
## - LAB       3   2331.5 2351.5
## - IMPEXAC   1   2927.5 2951.5
## - REGTEN    1   3843.6 3867.6
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
##
## Step:  AIC=2285.49
## cat2 ~ TAMAMU + DENSIDAD + SEXO + LAB + REGTEN + IMPEXAC
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
##            Df Deviance    AIC
## <none>          2265.5 2285.5
## - TAMAMU    1   2267.8 2285.8
```

```
## + ESTUD     3   2259.8 2285.8
## + SUPERF    1   2265.1 2287.1
## + EDAD      1   2265.5 2287.5
## - DENSIDAD  2   2273.4 2289.4
## - SEXO      1   2271.7 2289.7
## - LAB       3   2338.3 2352.3
## - IMPEXAC   1   3021.8 3039.8
## - REGTEN    1   3870.1 3888.1
##
## Call:  glm(formula = cat2 ~ TAMAMU + DENSIDAD + SEXO + LAB + REGTEN +
##     IMPEXAC, family = "binomial", data = datos_train_glm)
##
## Coefficients:
## (Intercept)      TAMAMU1    DENSIDAD2    DENSIDAD3        SEXO1
##     -7.6203      -0.2692      -0.3635      -0.3503      -0.2833
##        LAB2         LAB3         LAB4      REGTEN1      IMPEXAC
##     -1.7291      -2.0509      -1.2089       6.8968       0.3267
##
## Degrees of Freedom: 3240 Total (i.e. Null);  3231 Residual
## Null Deviance:       4487
## Residual Deviance: 2265  AIC: 2285
```

```
regresion_buena <- glm(cat2 ~ DENSIDAD + LAB + REGTEN + IMPEXAC, family ="binomial", data = datos_train
```

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

```
summary(regresion_buena)
```

```
##
## Call:
## glm(formula = cat2 ~ DENSIDAD + LAB + REGTEN + IMPEXAC, family = "binomial",
##     data = datos_train_glm)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9692  -0.4281  -0.0209   0.5908   3.3589
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -7.93927    0.38308 -20.725  < 2e-16 ***
## DENSIDAD2   -0.33310    0.13740  -2.424   0.0153 *
## DENSIDAD3   -0.18280    0.12080  -1.513   0.1302
## LAB2        -1.72168    0.34848  -4.940 7.79e-07 ***
## LAB3        -2.09304    0.28544  -7.333 2.26e-13 ***
## LAB4        -1.15632    0.18559  -6.231 4.65e-10 ***
## REGTEN1      6.87786    0.29484  23.328  < 2e-16 ***
## IMPEXAC      0.32102    0.01655  19.391  < 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: 4486.7  on 3240  degrees of freedom
## Residual deviance: 2273.5  on 3233  degrees of freedom
## AIC: 2289.5
```

```
##
## Number of Fisher Scoring iterations: 7
```

Primeramente definimos nuestro modelo predictivo conn todas las variables, depués mediante el step tanto fordward como backward nos arroja un modelo con otras variables, comparandiolos con el criterio de Akaike el que nos aporta un numero menor de AIC, es el que hemos conseguido con el Step, tanto por el método forward como el bacward.

```
datos_test_glm <- datos_test[,-11]
prediccion <- predict(regresion_buena, datos_test_glm, type = 'response')
```

Con la predicción hecha, el siguiente paso requiere que se clasifique cada valor de la predicción en 0 o 1. Para ello se va a averiguar cual es el mejor "cutoff" y posteriormente se realizará la predicción.

```
set.seed(1234)
searchgrid = seq(0.01, 1, 0.01)
result = cbind(searchgrid, NA)
cost1 <- function(r, pi){
  weight1 = 1
  weight0 = 1
  c1 = (r==1)&(pi<pcut) #logical vector - true if actual 1 but predict 0
  c0 = (r==0)&(pi>pcut) #logical vector - true if actual 0 but predict 1
  return(mean(weight1*c1+weight0*c0))
}

for(i in 1:length(searchgrid)) {
  pcut <- result[i,1]
  result[i,2] <- cv.glm(data = datos_train_glm, glmfit = regresion_buena, cost = cost1, K=5)$delta[2]
}

result[which.min(result[,2]),]
```

```
## searchgrid
##  0.5000000  0.1320607
```

El resultado es un intervaloi de confianza en el que se encuentra nuesto cutt of óptimo. El cutoff óptimo se encontrará entre 0.48 y 0.129. Se va a comprobar cual es el mejor. Después se representa la matriz de confusión para ver la precisión del modelo.

```
set.seed(1234)
matConf_glm <- table(ifelse(prediccion >= 0.48, 1, 0), datos_test_glm$cat2)
matConf_glm
```

```
##
##       0   1
##   0 338  42
##   1  72 359
```

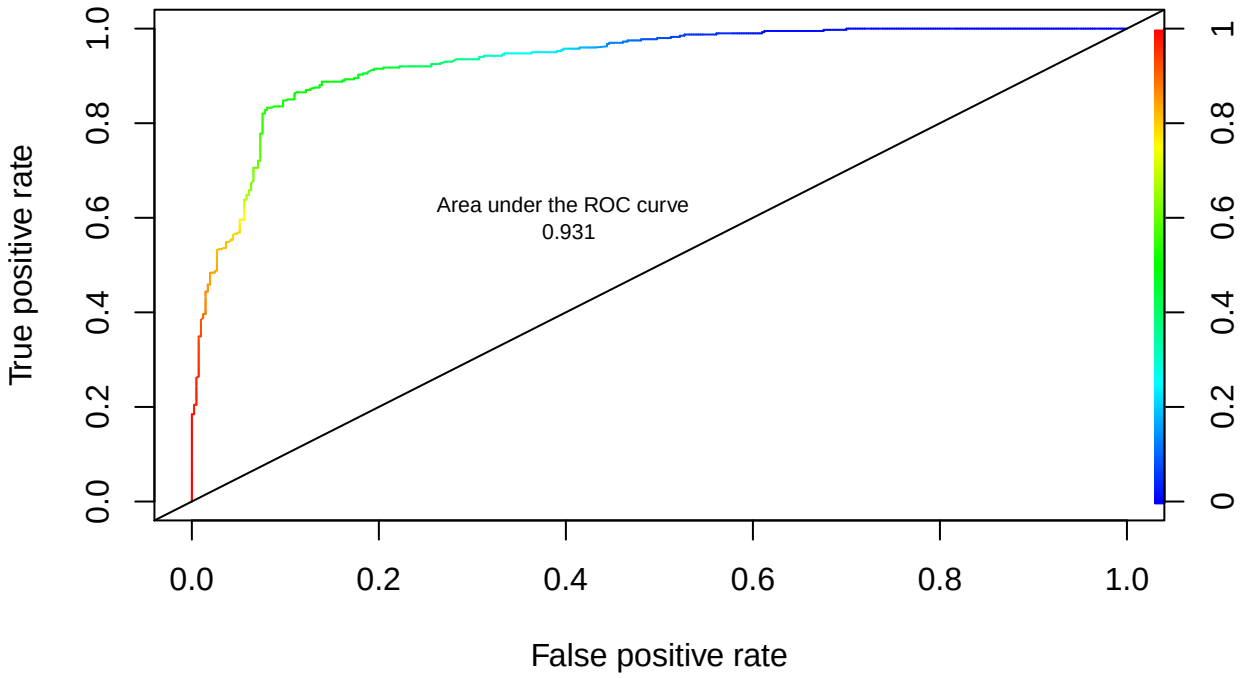
```
accuracy_glm <-sum(diag(matConf_glm))/sum(matConf_glm)
accuracy_glm
```

```
## [1] 0.8594328
```

Ahora se va a representar la curva ROC

```
set.seed(1234)
prediccion1 <- prediction(prediccion, datos_test$cat2)
AUC <- performance(prediccion1, "auc")
perf <- performance(prediccion1, "tpr", "fpr")
```

```r
plot(perf, colorize = TRUE) # Establecemos el color.
abline(a = 0, b = 1)
text(0.4, 0.6, paste(AUC@y.name, "\n", round(unlist(AUC@y.values), 3)), cex = 0.7)
```



Con un cutoff de 0.48, se obtiene un accuracy de 0.88 lo cual significa que el modelo de regresión logística es muy bueno. Además, el resultado que arroja la curva ROC es de 0.93, que es un resultado muy aceptable. La conclusión que se extrae es que las variables que mejor predicen el consumo de carne de vacuno en las familias son los ingresos familiares, la situación laboral y si paga o no hipoteca o alquiler. Es evidente que a mayor poder adquisitivo, más asequible es consumir este tipo de producto.

```r
rm(regresion)
rm(prediccion)
```

ÁRBOLES DE DECISIÓN

Un árbol de decisión es un mapa de los posibles resultados de una serie de decisiones relacionadas. Permite que un individuo o una organización comparen posibles acciones entre sí según sus costos, probabilidades y beneficios. Se pueden usar para dirigir un intercambio de ideas informal o trazar un algoritmo que anticipe matemáticamente la mejor opción.

```r
library(rpart)
set.seed(1234)
arbol <- rpart(cat2 ~ .,
               data=datos_train_glm,
               method="class",
               parms=list(split="information"))

arbol.pred1 <- predict(arbol, datos_test_glm, type="class")
```

```
tabla.clasif.arbol1 <- table(datos_test_glm$cat2, arbol.pred1,
                             dnn=c("Actual", "Predicted"))
print(arbol)
```

```
## n= 3241
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 3241 1549 0 (0.52206109 0.47793891)
##   2) REGTEN=0 1049   94 0 (0.91039085 0.08960915)
##     4) IMPEXAC< 23.7 970   31 0 (0.96804124 0.03195876) *
##     5) IMPEXAC>=23.7 79   16 1 (0.20253165 0.79746835) *
##   3) REGTEN=1 2192  737 1 (0.33622263 0.66377737)
##     6) IMPEXAC< 7.435 604   85 0 (0.85927152 0.14072848) *
##     7) IMPEXAC>=7.435 1588  218 1 (0.13727960 0.86272040) *
```

```
tabla.clasif.arbol1
```

```
##       Predicted
## Actual   0   1
##      0 351  59
##      1  37 364
```

Aquí se representa el accuracy del arbol sin podar. Más tarde lo podaremos para comprobar su estabilidad.

```
tcc2 <- 100 * sum(diag(tabla.clasif.arbol1))/sum(tabla.clasif.arbol1)
tcc2
```

```
## [1] 88.16276
```

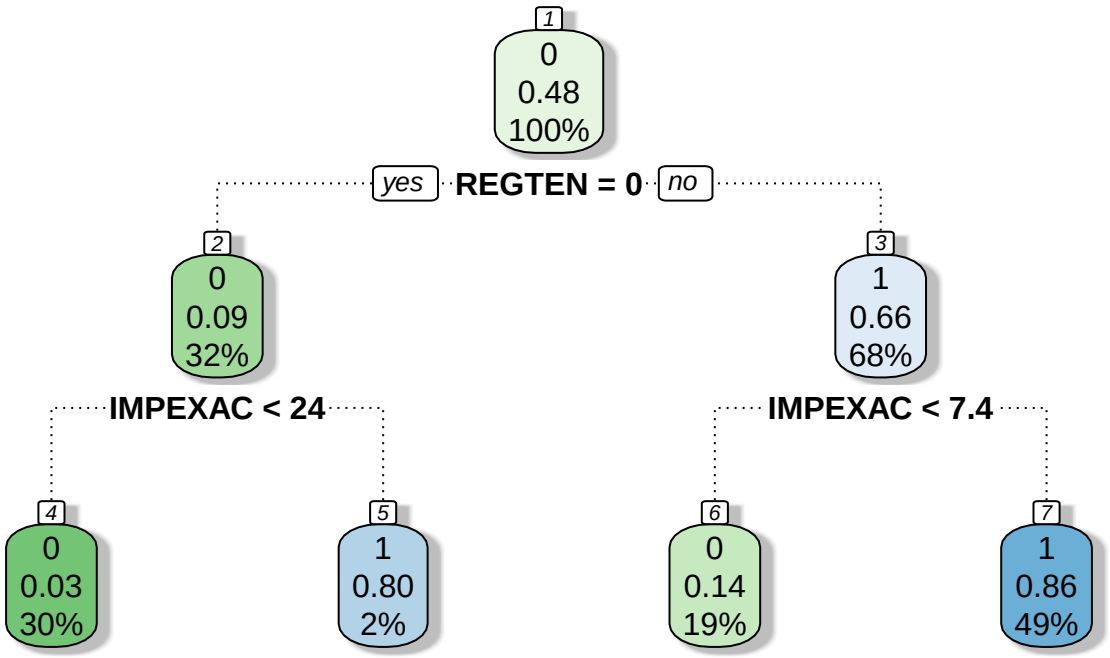
Se observa que el accuracy del árbol es del 89.07%, lo cual es un accuracy buenísimo.

Aquí se representa la importancia de las variables para la construcción del árbol y el gráfico del mismo.

```
set.seed(1234)
arbol$variable.importance
```

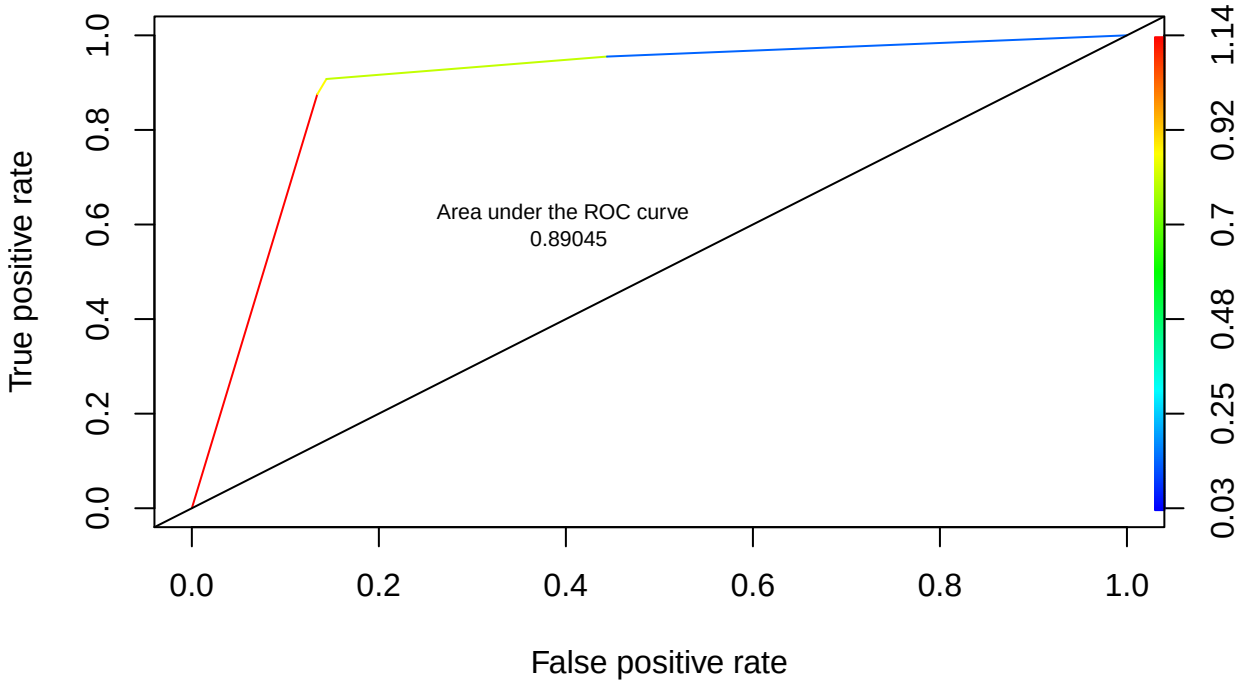
```
##   IMPEXAC    REGTEN       LAB      EDAD    SUPERF
## 659.87303 527.33377 177.93928 153.82663  37.84051
```

```
rpart.plot(arbol, box.palette = "GnBu", branch.lty = 3,
           shadow.col = "gray",
           nn = TRUE, main = "Árbol de clasificación sin podar")
```



Cada uno de los cuadrados coloreados representa un nodo de nuestro ?rbol, con su regla de clasificaci?n correspondiente, Cada nodo est? coloreado de acuerdo a la categor?a mayoritaria entre los datos que agrupa. Esta es la categor?a que ha predicho el modelo para ese grupo. Para los cortes de los nodos se ha llevado a cabo un criterio del par?metro de complejidad (CP), el cual, el par?metro de complejidad no es el error en ese nodo en particular. Es la cantidad por la cual la divisi?n de ese nodo mejor? el error relativo. El CP del siguiente nodo es solo 0.01 (que es el l?mite predeterminado para decidir cu?ndo considerar las divisiones) Nosotros para podar el arbol elegiremos uno superior. Ahora se va a representar la curva ROC.

```
set.seed(1234)
prediccion_arbol <- predict(arbol, datos_test, type="prob")[,2]
pred_arbol = prediction(prediccion_arbol, datos_test$cat2)
AUC <- performance(pred_arbol, "auc")
perf1 <- performance(pred_arbol, "tpr", "fpr")
plot(perf1, colorize = TRUE)
abline(a = 0, b = 1)
text(0.4, 0.6, paste(AUC@y.name, "\n", round(unlist(AUC@y.values), 5)), cex = 0.7)
```



El resultado de la curva ROC es del 90%, lo cual es un resultado muy bueno. Seguidamente se va a realizar la poda del árbol. Para ello se calculará el mejor CP (parámetro de complejidad relativo error mínimo) y posteriormente se podará. Se podrá comprobar si los resultados del árbol podado son mejores o peores que los del árbol sin podar.

```
set.seed(1234)
arbol$cptable[which.min(arbol$cptable[,"xerror"]),"CP"]
```

```
## [1] 0.01
```

```
printcp(arbol)
```

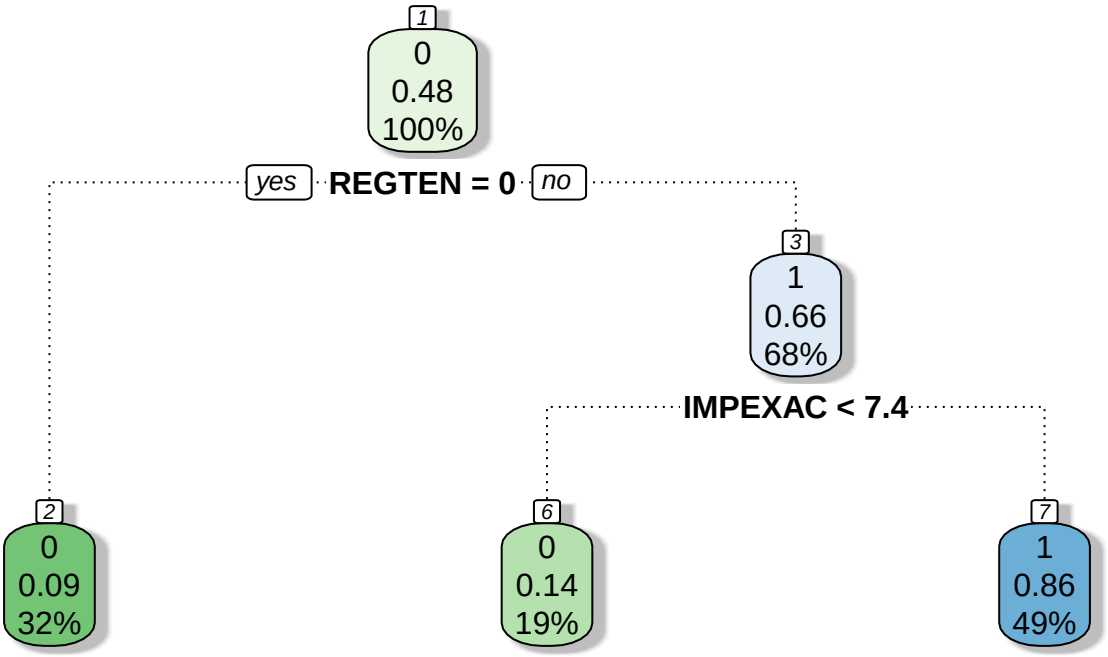
```
##
## Classification tree:
## rpart(formula = cat2 ~ ., data = datos_train_glm, method = "class",
##     parms = list(split = "information"))
##
## Variables actually used in tree construction:
## [1] IMPEXAC REGTEN
##
## Root node error: 1549/3241 = 0.47794
##
## n= 3241
##
##         CP nsplit rel error  xerror     xstd
## 1 0.463525      0   1.00000 1.00000 0.018358
## 2 0.280181      1   0.53648 0.53648 0.016048
## 3 0.030342      2   0.25629 0.25888 0.012102
```

```
## 4 0.010000       3   0.22595 0.23047 0.011506
```

```
arbol_podado <- prune(arbol, cp = 0.22914)
```

Ahora se va a calcular, igual que para el árbol anterior, la predicción pertinente con su posterior precisión o accuracy.

```
set.seed(1234)
rpart.plot(arbol_podado, box.palette = "GnBu", branch.lty = 3,
           shadow.col = "gray",
           nn = TRUE, main = "Árbol de clasificación podado")
```



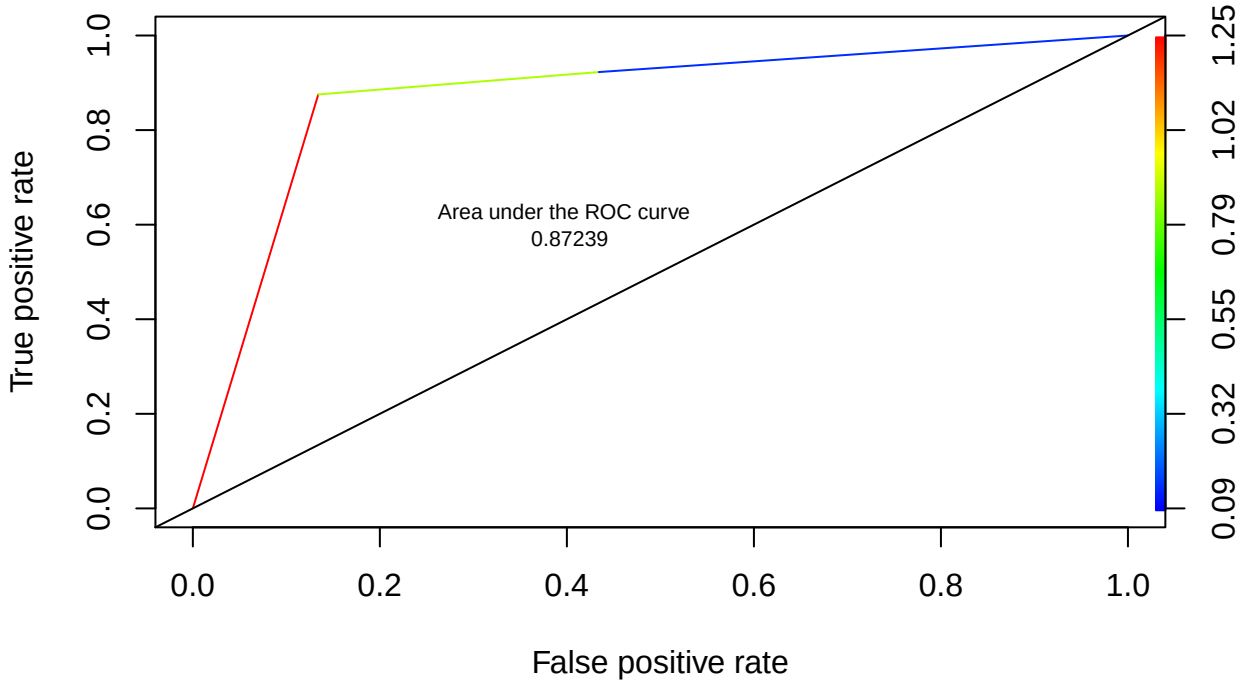

```
arbol_prediccion <- predict(arbol_podado, datos_test_glm, type = "class")
arbol_resultado_total <- table(datos_test_glm$cat2, arbol_prediccion,
                               dnn = c("Actual", "Predicted"))

tcc1 <- 100 * sum(diag(arbol_resultado_total))/sum(arbol_resultado_total)
tcc1
```

```
## [1] 87.05302
```

El accuracy del árbol podado es del 87.74%, este resultado es muy bueno también; sin embargo, no alcanza el resultado del accuracy del árbol sin podar, así que se aceptará como mejor resultado el del árbol sin podar. A continuación, se representa la curva ROC de este árbol.

```
set.seed(1234)
prediccion_arbol2 <- predict(arbol_podado, datos_test, type="prob")[,2]
pred_arbol2 = prediction(prediccion_arbol2, datos_test$cat2)
AUC2 <- performance(pred_arbol2, "auc")
```

```
perf2 <- performance(pred_arbol2, "tpr", "fpr")
plot(perf2, colorize = TRUE)
abline(a = 0, b = 1)
text(0.4, 0.6, paste(AUC2@y.name, "\n", round(unlist(AUC2@y.values), 5)), cex = 0.7)
```



Como era de esperar, el resultado de esta curva ROC es del 88%, inferior al anterior, por lo que damos por mejor arbol, el que está sin podar.

```
rm(prediccion_arbol)
rm(prediccion_arbol2)
rm(prediccion1)
rm(prediccion)
```

```
## Warning in rm(prediccion): objeto 'prediccion' no encontrado
```

```
rm(arbol)
rm(arbol_podado)
rm(arbol_resultado_total)
```

Modelo de Árboles para *Cat3*

Seguiremos el mismo procedimiento que para *cat2*

```
datos_train_arbol3 <- datos_train[,-10]
datos_test_arbol3 <- datos_test[,-10]

set.seed(1234)
```

```
arbol3 <- rpart(cat3 ~ .,
                data=datos_train_arbol3,
                method="class",
                parms=list(split="information"))

arbol.pred3 <- predict(arbol3, datos_test_arbol3, type="class")

tabla.clasif.arbol3 <- table(datos_test_arbol3$cat3, arbol.pred3,
                             dnn=c("Actual", "Predicted"))
print(arbol3)
```

```
## n= 3241
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 3241 2115 2 (0.34464671 0.34742363 0.30792965)
##    2) REGTEN=0 1049  317 1 (0.69780744 0.25548141 0.04671115)
##      4) IMPEXAC< 14.765 676   45 1 (0.93343195 0.06360947 0.00295858) *
##      5) IMPEXAC>=14.765 373  148 2 (0.27077748 0.60321716 0.12600536)
##       10) IMPEXAC< 30.19 341  118 2 (0.29618768 0.65395894 0.04985337) *
##       11) IMPEXAC>=30.19 32    2 3 (0.00000000 0.06250000 0.93750000) *
##    3) REGTEN=1 2192 1243 3 (0.17563869 0.39142336 0.43293796)
##      6) IMPEXAC< 11.44 1308  551 2 (0.26758410 0.57874618 0.15366972)
##       12) IMPEXAC< 7.425 589  285 1 (0.51612903 0.36672326 0.11714771)
##         24) IMPEXAC< 5.94 260   71 1 (0.72692308 0.20769231 0.06538462) *
##         25) IMPEXAC>=5.94 329  167 2 (0.34954407 0.49240122 0.15805471) *
##       13) IMPEXAC>=7.425 719  178 2 (0.06397775 0.75243394 0.18358832) *
##      7) IMPEXAC>=11.44 884  136 3 (0.03959276 0.11425339 0.84615385) *
```

```
tabla.clasif.arbol3
```

```
##       Predicted
## Actual   1   2   3
##      1 206  62   5
##      2  21 237  32
##      3   6  46 196
```

```
tcc3 <- 100 * sum(diag(tabla.clasif.arbol3))/sum(tabla.clasif.arbol3)
tcc3
```

```
## [1] 78.79162
```

Se observa que el accuracy del árbol es del 78.79%, lo cual es un accuracy razonablemente bueno, pero inferior a lo esperado.

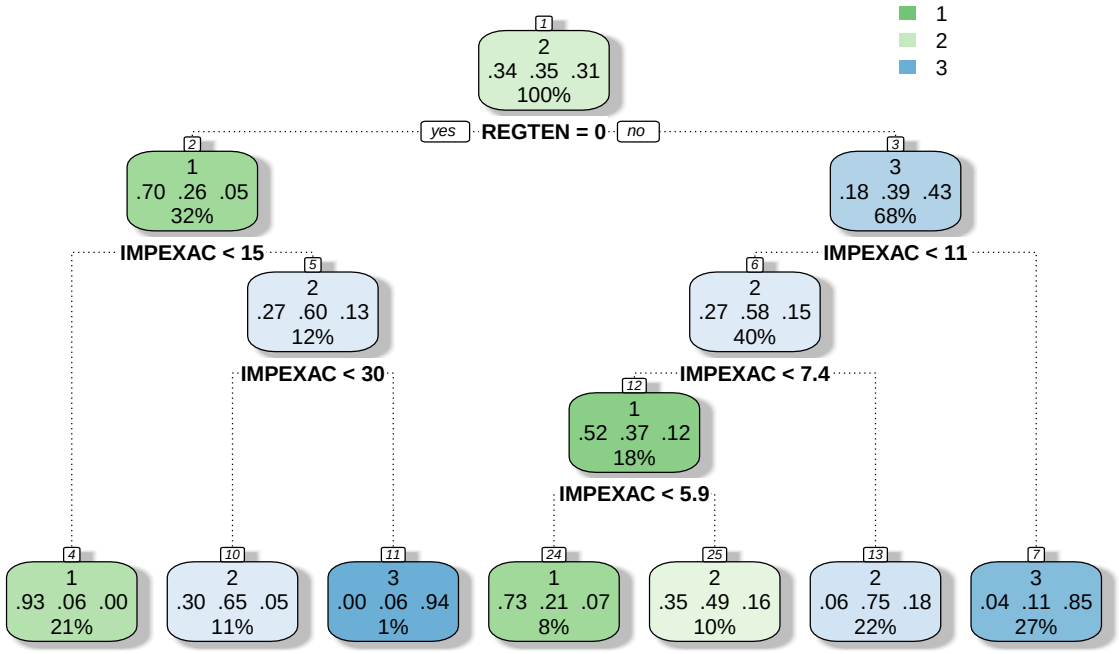
Aquí se representa la importancia de las variables para la construcción del árbol y el gráfico del mismo.

```
arbol3$variable.importance
```

```
##     IMPEXAC      REGTEN         LAB       ESTUD        EDAD      SUPERF
## 1115.642350  507.712687  284.103848  206.616282  197.677338   46.332761
##        SEXO
##    6.390798
```

```
rpart.plot(arbol3, box.palette = "GnBu", branch.lty = 3,
           shadow.col = "gray",
           nn = TRUE, main = "Árbol de clasificación sin podar")
```

Árbol de clasificación sin podar




```
library(caret)
arbol.pred_2 <- predict(arbol3, datos_test_arbol3, type="class")

tabla.clasif.arbol2 <- table(datos_test_arbol3$cat3, arbol.pred_2,
                             dnn=c("Actual", "Predicted"))

confusionMatrix(as.factor(arbol.pred_2),datos_test_arbol3$cat3)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   2   3
##          1 206  21   6
##          2  62 237  46
##          3   5  32 196
##
## Overall Statistics
##
##                Accuracy : 0.7879
##                  95% CI : (0.7581, 0.8156)
##     No Information Rate : 0.3576
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6803
##  Mcnemar's Test P-Value : 4.326e-05
##
```

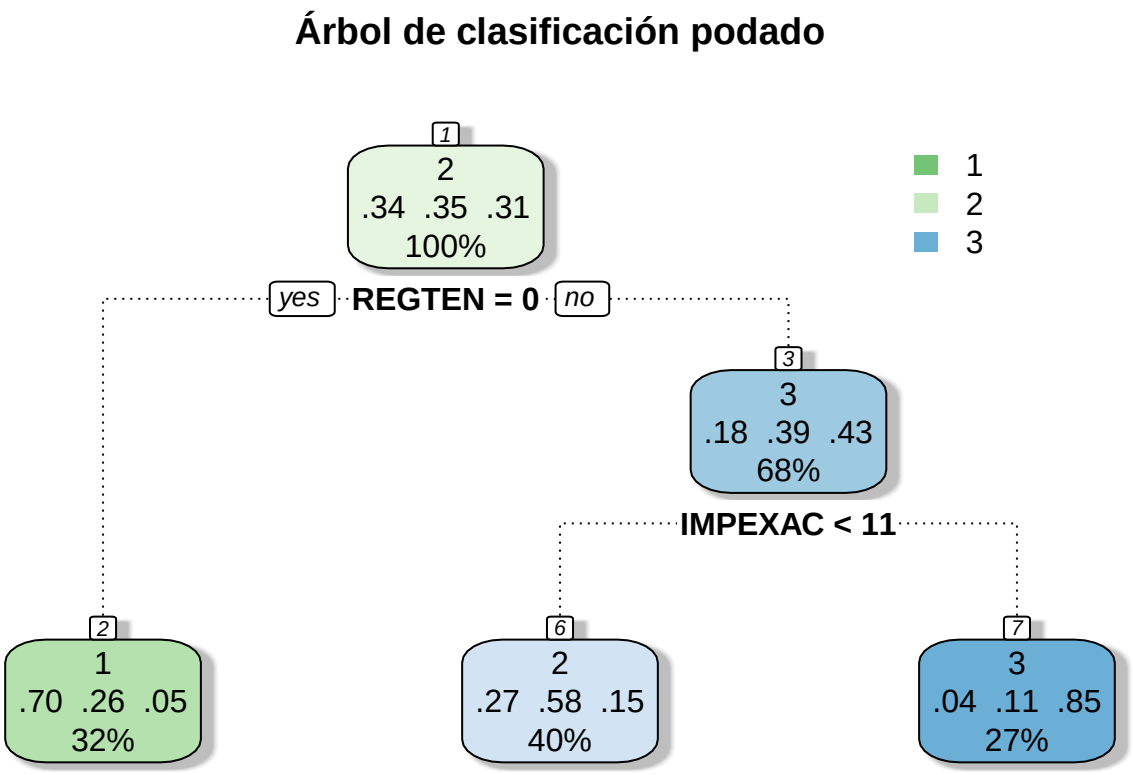
```
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3
## Sensitivity            0.7546   0.8172   0.7903
## Specificity            0.9498   0.7927   0.9343
## Pos Pred Value         0.8841   0.6870   0.8412
## Neg Pred Value         0.8841   0.8863   0.9100
## Prevalence             0.3366   0.3576   0.3058
## Detection Rate         0.2540   0.2922   0.2417
## Detection Prevalence   0.2873   0.4254   0.2873
## Balanced Accuracy      0.8522   0.8050   0.8623
```

```
tcc3 <- 100 * sum(diag(tabla.clasif.arbol2))/sum(tabla.clasif.arbol2)
tcc3
```

```
## [1] 78.79162
```

La curva ROC no se puede realizar ya que solo permite clasificaciones binarias y no estamos antes una variable de tal tipo.

Seguidamente se realizará la poda como se ha hecho anteriormente, eligiendo para ello el parámetro de complejidad correcto. Comprobaremos si los resultados obtenidos son mejores o peores que con el árbol original.



Comprobamos nuevamente con los datos de test la precisión del modelo de árbol ded clasificación para Categoria3.

```
##        Predicted
## Actual   1   2   3
```

```
##      1 177  91   5
##      2  71 188  31
##      3  15  42 191
```

```
## [1] 68.55734
```