Proyecto - Aprendizaje supervisado con caret

Tomas Lemus

20/11/2020

- Analisis para sexo femenino
- REGRESION
    - modelo de regresión simple y polinomico.

```
##install.packages("readr")
#install.packages("tidyverse")
#install.packages("webshot")
```

```
df = read_csv("Alzheimer.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_double(),
##   SEX = col_character(),
##   CLASS = col_character()
## )
## i Use `spec()` for the full column specifications.
```

```
head(df)
```

AGE <dbl>	SEX <chr>	BRAIN_VOL… <dbl>	GM_VO… <dbl>	WM_VO… <dbl>	CSF_VOL… <dbl>	GM_BRAIN_QUOTI… <dbl>	WM_BR/
68.00	MALE	1410.726	598.9312	433.2638	378.5312	0.424555	
82.94	FEMALE	1367.844	551.8513	445.8658	370.1266	0.403446	
72.85	FEMALE	1153.635	443.9688	393.1218	316.5447	0.384843	
73.00	FEMALE	1546.814	664.6387	492.1400	390.0350	0.429682	
71.12	FEMALE	1325.557	558.7274	433.0302	333.7991	0.421504	
78.13	MALE	1399.918	526.0841	441.3189	432.5154	0.375796	

6 rows | 1-8 of 228 columns

```
print(dim(df))
```

```
## [1] 262 228
```

Brevemente podemos observar sujetos masculinos y femeninos, desde 55 años de edad con una media de 74, además de variables que representan medidas volumétricas del cerebro (volumen y grosor de regiones anatómicas cerebrales) para personas sanas o con Alzheimer determinadas por la categoria CLASS.Este dataset contiene inicialmente 262 registros con 228 variables entre los cuales seleccionaremos segun sea necesario.

```
select(df,CLASS,BRAIN_VOLUME,SEX)
```

CLASS <chr>	BRAIN_VOLUME <dbl>	SEX <chr>
AD	1410.726	MALE
AD	1367.844	FEMALE
AD	1153.635	FEMALE
AD	1546.814	FEMALE
AD	1325.557	FEMALE
AD	1399.918	MALE
AD	1241.322	FEMALE
AD	1514.051	FEMALE
AD	1238.210	FEMALE
AD	1512.991	MALE

1-10 of 262 rows Previous **1** 2 3 4 5 6 ... 27 Next

```
df=select(df,CLASS,BRAIN_VOLUME,SEX)
df_H=df%>% filter(CLASS=="HEALTHY",SEX=="MALE")
df_A=df%>% filter(CLASS=="AD",SEX=="MALE")
head(df_H)
```

CLASS <chr>	BRAIN_VOLUME <dbl>	SEX <chr>
HEALTHY	1607.759	MALE
HEALTHY	1348.900	MALE
HEALTHY	1390.601	MALE
HEALTHY	1519.151	MALE
HEALTHY	1601.988	MALE
HEALTHY	1456.863	MALE

6 rows

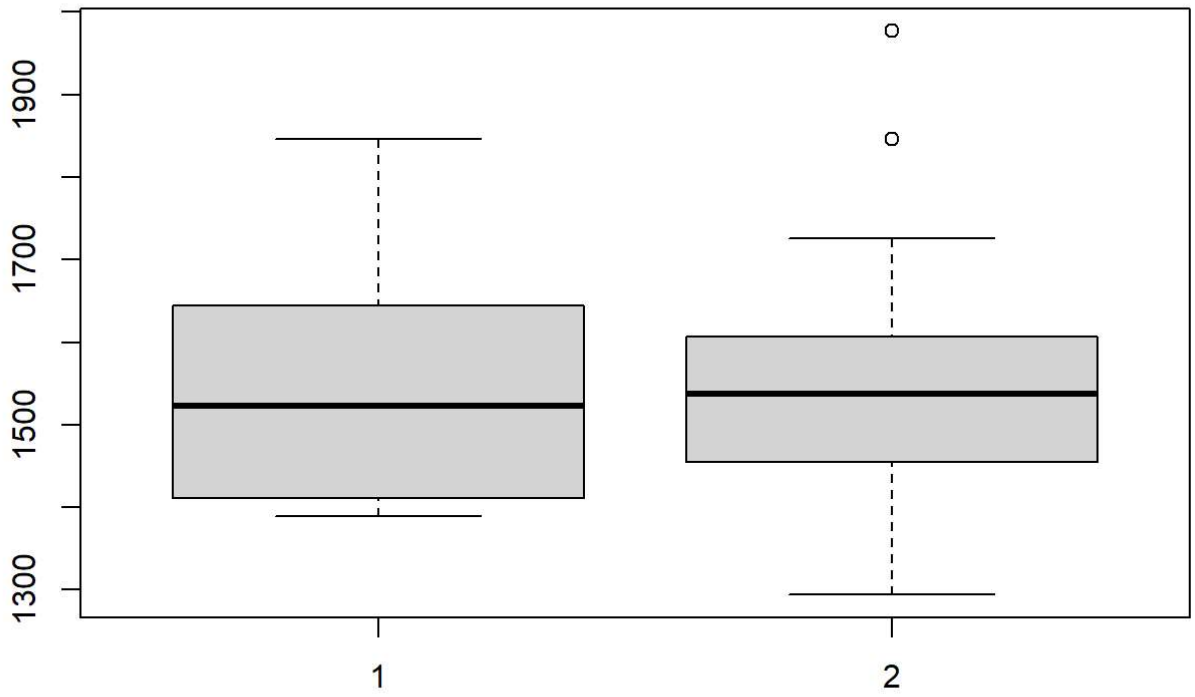
CONTRASTE DE HIPÓTESIS: Se supone normalidad y homocedasticidad u homogeneidad de varianzas, podemos realizar nuestro contraste. ha: μAD<μH la variable volumen cerebral es significativamente menor en sujetos con Alzheimer que en sujetos sanos.(<) ho: μAD >= μH volumen cerebral de AD no es menor que en sanos

```
t.test(df_A$BRAIN_VOLUME,df_H$BRAIN_VOLUME, alternative = "less")
```

```
##
##  Welch Two Sample t-test
##
## data:  df_A$BRAIN_VOLUME and df_H$BRAIN_VOLUME
## t = 0.023782, df = 14.19, p-value = 0.5093
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##      -Inf 74.43254
## sample estimates:
## mean of x mean of y
##  1542.332  1541.340
```

No se encontró evidencia suficiente para descartar H0, por lo tanto se estima que el volumen cerebral de los sujetos AD no es menos que aquellos sanos de sexo masculino.

```
boxplot(df_A$BRAIN_VOLUME,df_H$BRAIN_VOLUME)
```



Proceso para sexo femenino:

```
dFH=df%>% filter(CLASS=="HEALTHY",SEX=="FEMALE")
dFA=df%>% filter(CLASS=="AD",SEX=="FEMALE")
```

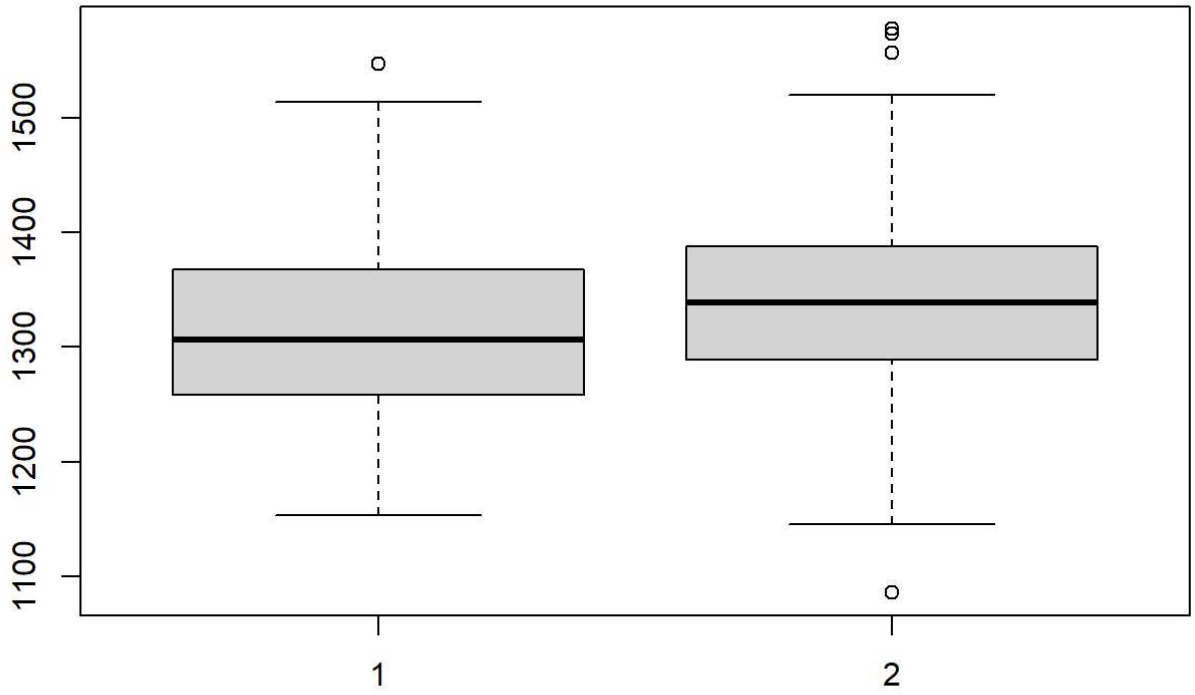
ha: μAD<μH la variable volumen cerebral es significativamente menor en sujetos con Alzheimer que en sujetos sanos de sexo femenino.(<) ho: μAD >= μH volumen cerebral de AD no es menor que en sanos de sexo feminino.

```
t.test(dFA$BRAIN_VOLUME,dFH$BRAIN_VOLUME, alternative = "less")
```

```
##
##  Welch Two Sample t-test
##
## data:  dFA$BRAIN_VOLUME and dFH$BRAIN_VOLUME
## t = -0.64776, df = 19.635, p-value = 0.2623
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##      -Inf 28.21545
## sample estimates:
## mean of x mean of y
##  1322.434  1339.380
```

No se encontró evidencia suficiente para descartar H0, por lo tanto se estima que el volumen cerebral de los sujetos AD no es menos que aquellos sanos de sexo femenino.

```
boxplot(dFA$BRAIN_VOLUME,dFH$BRAIN_VOLUME)
```



Repetir el análisis anterior, pero para comprobar si las variables de volumen de sustancia gris o de sustancia blanca son diferentes entre sujetos sanos y sujetos con AD (realizar un contraste por cada variable).

```
#masculinos
df2 = read_csv("Alzheimer.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_double(),
##   SEX = col_character(),
##   CLASS = col_character()
## )
## i Use `spec()` for the full column specifications.
```

```
dfAM= df2 %>% select(SEX,CLASS,GM_VOLUME,WM_VOLUME) %>% filter(SEX=="MALE",CLASS=="AD")
dfHM= df2 %>% select(SEX,CLASS,GM_VOLUME,WM_VOLUME) %>% filter(SEX=="MALE",CLASS=="HEALTHY")
head(dfAM)
```

SEX <chr>	CLASS <chr>	GM_VOLUME <dbl>	WM_VOLUME <dbl>
MALE	AD	598.9312	433.2638
MALE	AD	526.0841	441.3189
MALE	AD	580.8741	515.4593
MALE	AD	601.7287	436.1062
MALE	AD	694.6025	497.4525
MALE	AD	643.6562	519.6712

6 rows

CONTRASTE DE HIPÓTESIS:

ha: μADG!=μADW las variables de volumen de sustancia gris o de sustancia blanca son diferentes entre sujetos con AD ho: μADG = μADW las variables de volumen de sustancia gris o de sustancia blanca no son diferentes entre sujetos con AD

```
t.test(dfAM$GM_VOLUME,dfAM$WM_VOLUME, alternative = "two.sided")
```

```
##
##  Welch Two Sample t-test
##
## data:  dfAM$GM_VOLUME and dfAM$WM_VOLUME
## t = 6.213, df = 18.012, p-value = 7.285e-06
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  107.0281 216.3849
## sample estimates:
## mean of x mean of y
##  649.8218  488.1153
```

Valor p menor que 0.05 por lo tanto hay evidencia suficiente para rechazar h0, y se estima diferencia entre las sustancias blancas y gris para sujetos con AD masculinos.

```
head(dfHM)
```

SEX <chr>	CLASS <chr>	GM_VOLUME <dbl>	WM_VOLUME <dbl>
MALE	HEALTHY	701.2512	493.5096
MALE	HEALTHY	557.7522	428.0321
MALE	HEALTHY	605.7468	408.6448
MALE	HEALTHY	666.9332	470.9092
MALE	HEALTHY	694.2115	520.7440
MALE	HEALTHY	625.9764	505.7925

6 rows

CONTRASTE DE HIPÓTESIS:

ha: μHG!=μHW las variables de volumen de sustancia gris o de sustancia blanca son diferentes entre sujetos sanos. ho: μHG = μHW las variables de volumen de sustancia gris o de sustancia blanca no son diferentes entre sujetos sanos.

```
t.test(dfHM$GM_VOLUME,dfHM$WM_VOLUME, alternative = "two.sided")
```

```
##
##  Welch Two Sample t-test
##
## data:  dfHM$GM_VOLUME and dfHM$WM_VOLUME
## t = 20.271, df = 187.88, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  146.9420 178.6243
## sample estimates:
## mean of x mean of y
##  664.1903  501.4072
```

Valor p menor que 0.05 por lo tanto hay evidencia suficiente para rechazar h0, y se estima diferencia entre las sustancias blancas y gris para sujetos sanos de sexo masculino.

Analisis para sexo femenino

```
dfAF= df2 %>% select(SEX,CLASS,GM_VOLUME,WM_VOLUME) %>% filter(SEX=="FEMALE",CLASS=="AD")
dfHf= df2 %>% select(SEX,CLASS,GM_VOLUME,WM_VOLUME) %>% filter(SEX=="FEMALE",CLASS=="HEALTHY"
)
head(dfAF)
```

SEX <chr>	CLASS <chr>	GM_VOLUME <dbl>	WM_VOLUME <dbl>
FEMALE	AD	551.8513	445.8658
FEMALE	AD	443.9688	393.1218
FEMALE	AD	664.6387	492.1400
FEMALE	AD	558.7274	433.0302

SEX <chr>	CLASS <chr>	GM_VOLUME <dbl>	WM_VOLUME <dbl>
FEMALE	AD	409.4508	437.0235
FEMALE	AD	614.5850	502.2800

6 rows

CONTRASTE DE HIPÓTESIS:

ha: μADG!=μADW las variables de volumen de sustancia gris o de sustancia blanca son diferentes entre sujetos con AD de sexo femenino. ho: μADG = μADW las variables de volumen de sustancia gris o de sustancia blanca no son diferentes entre sujetos con AD de sexo femenino.

```
t.test(dfAF$GM_VOLUME,dfAF$WM_VOLUME, alternative = "two.sided")
```

```
##
##  Welch Two Sample t-test
##
## data:  dfAF$GM_VOLUME and dfAF$WM_VOLUME
## t = 7.4354, df = 23.426, p-value = 1.31e-07
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  102.7864 181.9164
## sample estimates:
## mean of x mean of y
##  567.9276  425.5762
```

Valor p menor que 0.05 por lo tanto hay evidencia suficiente para rechazar h0, y se estima diferencia entre las sustancias blancas y gris para sujetos con AD de sexo femenino.

```
head(dfHf)
```

SEX <chr>	CLASS <chr>	GM_VOLUME <dbl>	WM_VOLUME <dbl>
FEMALE	HEALTHY	545.2222	365.4857
FEMALE	HEALTHY	624.2940	413.8128
FEMALE	HEALTHY	611.3537	466.5913
FEMALE	HEALTHY	606.0396	470.5016
FEMALE	HEALTHY	520.4438	371.9925
FEMALE	HEALTHY	574.9925	437.9700

6 rows

CONTRASTE DE HIPÓTESIS:

ha: μHG!=μHW las variables de volumen de sustancia gris o de sustancia blanca son diferentes entre sujetos sanos de sexo femenino. ho: μHG = μHW las variables de volumen de sustancia gris o de sustancia blanca no son diferentes entre sujetos sanos de sexo femenino.

```
t.test(dfHf$GM_VOLUME,dfHf$WM_VOLUME, alternative = "two.sided")
```

```
##
##  Welch Two Sample t-test
##
## data:  dfHf$GM_VOLUME and dfHf$WM_VOLUME
## t = 31.268, df = 231.26, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  160.9704 182.6209
## sample estimates:
## mean of x mean of y
##  600.2010  428.4054
```

Valor p menor que 0.05 por lo tanto hay evidencia suficiente para rechazar h0, y se estima diferencia entre las sustancias blancas y gris para sujetos sanos de sexo femenino.

¿Es diferente la incidencia de la enfermedad en pacientes de distinto sexo?

¿Son independientes las variables SEX y CLASS? Usa un test χ2 para comprobar la independencia y la correlación entre ambas.

H0: Son independientes las variables SEX y CLASS. H1: No son independientes las variables SEX y CLASS.

```
tab <- table(df$SEX, df$CLASS)
tab
```

```
##
##           AD HEALTHY
##   FEMALE  17     137
##   MALE    13      95
```

```
chisq.test(tab)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  tab
## X-squared = 0.0027726, df = 1, p-value = 0.958
```

Valor p mayor que 0.05 por lo tanto no hay evidencia suficiente para rechazar h0, y se estima independencia entre las variables SEX y CLASS.

El índice χ2 toma el valor 0 cuando dos variables son independientes.

REGRESION

Nos planteamos establecer un patrón de atrofia anual (es decir, dependiendo de la edad) en el cerebro. ¿Cómo varía el volumen del cerebro con la edad?

Queremos establecer un modelo de regresión que use como predictores las variables sexo y edad para estimar el valor del volumen cerebral (variable BRAIN_VOLUME).

modelo de regresión simple y polinomico.

Hipotesis para modelos de regresión Ho: La función corresponde el modelo H1: La función no corresponde al modelo Hipotesis para anova Ho: El modelo más complejo no mejora al más simple H1: El modelo más complejo mejora al más simple

```
dfR= df2 %>% select(BRAIN_VOLUME,AGE,SEX)
modelo_1 <- lm(BRAIN_VOLUME ~ AGE, data = dfR)
modelo_2 <- lm(BRAIN_VOLUME ~ poly(AGE, 2), data = dfR)
modelo_3 <- lm(BRAIN_VOLUME ~ poly(AGE, 3), data = dfR)
modelo_4 <- lm(BRAIN_VOLUME ~ poly(AGE, 4), data = dfR)
modelo_5 <- lm(BRAIN_VOLUME ~ poly(AGE, 5), data = dfR)

anova(modelo_1, modelo_2, modelo_3, modelo_4, modelo_5)
```

	Res.Df <dbl>	RSS <dbl>	Df <dbl>	Sum of Sq <dbl>	F <dbl>	Pr(>F) <dbl>
1	260	5547943	*NA*	*NA*	*NA*	*NA*
2	259	5381389	1	166553.661	8.01507802	0.005007285
3	258	5379546	1	1842.884	0.08868526	0.766096807
4	257	5329182	1	50364.056	2.42367438	0.120749668
5	256	5319691	1	9491.478	0.45675930	0.499752488

5 rows

```
f1= lm(dfR$BRAIN_VOLUME ~ dfR$AGE + dfR$SEX)
f2= lm(dfR$BRAIN_VOLUME ~ dfR$AGE + dfR$SEX + I(dfR$AGE^2))
f3= lm(dfR$BRAIN_VOLUME ~ dfR$AGE + dfR$SEX + I(dfR$AGE^2) + I(dfR$AGE^3))
f4= lm(dfR$BRAIN_VOLUME ~ dfR$AGE + dfR$SEX + I(dfR$AGE^2) + I(dfR$AGE^3) + I(dfR$AGE^4))
f5= lm(dfR$BRAIN_VOLUME ~ dfR$AGE + dfR$SEX + I(dfR$AGE^2) + I(dfR$AGE^3) + I(dfR$AGE^4) + I
(dfR$AGE^5))
anova(f1,f2,f3,f4,f5)
```

	Res.Df <dbl>	RSS <dbl>	Df <dbl>	Sum of Sq <dbl>	F <dbl>	Pr(>F) <dbl>
1	259	2906017	*NA*	*NA*	*NA*	*NA*
2	258	2861992	1	44025.531	4.021707	0.04597483
3	257	2846303	1	15688.904	1.433172	0.23235911
4	256	2794262	1	52040.272	4.753849	0.03014739
5	255	2791479	1	2783.422	0.254264	0.61452396

5 rows

#degree 2 is the better

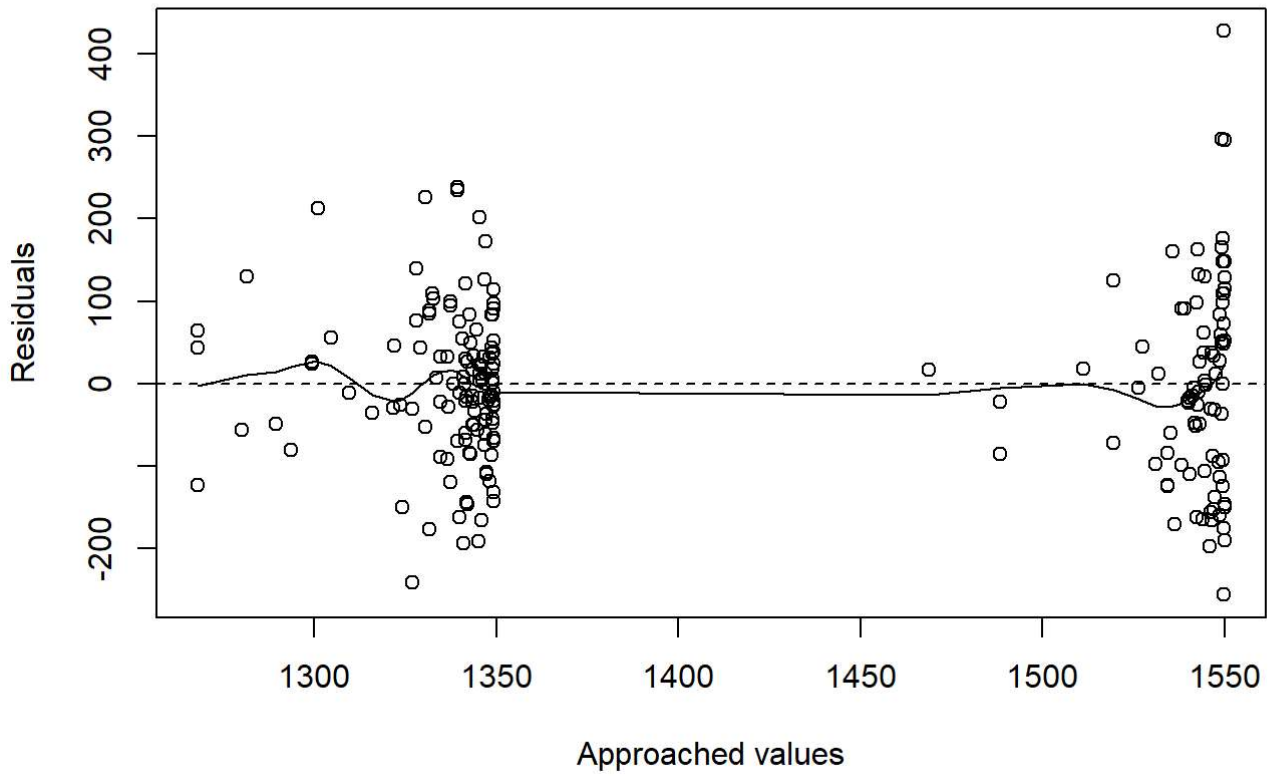
```
summary(f2)
```

```
##
## Call:
## lm(formula = dfR$BRAIN_VOLUME ~ dfR$AGE + dfR$SEX + I(dfR$AGE^2))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -255.59  -67.48    0.14   51.98  427.06
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  422.95693  418.29806   1.011   0.3129
## dfR$AGE       23.70259   11.31683   2.094   0.0372 *
## dfR$SEXMALE  200.74167   13.32028  15.070   <2e-16 ***
## I(dfR$AGE^2)  -0.15162    0.07611  -1.992   0.0474 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 105.3 on 258 degrees of freedom
## Multiple R-squared:  0.4861, Adjusted R-squared:  0.4801
## F-statistic: 81.34 on 3 and 258 DF,  p-value: < 2.2e-16
```

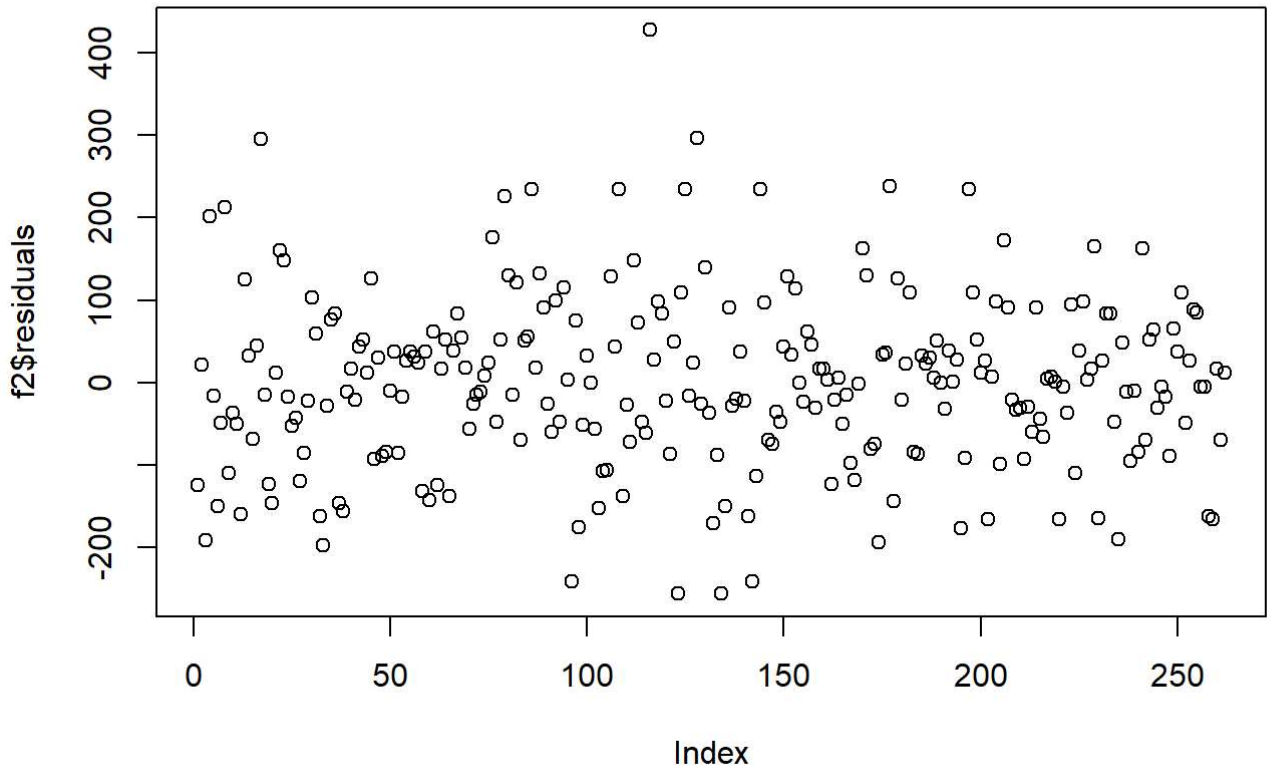
Cada una de las pendientes de un modelo de regresión lineal múltiple se define del siguiente modo: Si el resto de variables se mantienen constantes, por cada unidad que aumenta el predictor en cuestión, la variable Y varía en promedio tantas unidades como indica la pendiente. En el caso del predictor AGE, si el resto de variables no varían, por cada unidad de AGE que aumenta el volumen cerebral se aumenta en promedio 23.70259 unidades. Las variables son significativas (al menos '*' 0.05), El coeficiente de determinacion R^2(0.4982 o 49.82%): es el porcentaje de la variación en la variable de respuesta que es explicado por un modelo lineal. El R-cuadrado siempre está entre 0 y 100%:

0% indica que el modelo no explica ninguna porción de la variabilidad de los datos de respuesta en torno a su media. 100% indica que el modelo explica toda la variabilidad de los datos de respuesta en torno a su media.

```
plot.new()
plot(fitted(f2), residuals(f2), xlab = "Approached values",  ylab = "Residuals")
  abline(h=0, lty=2)
    lines(smooth.spline(fitted(f2), residuals(f2)))
```



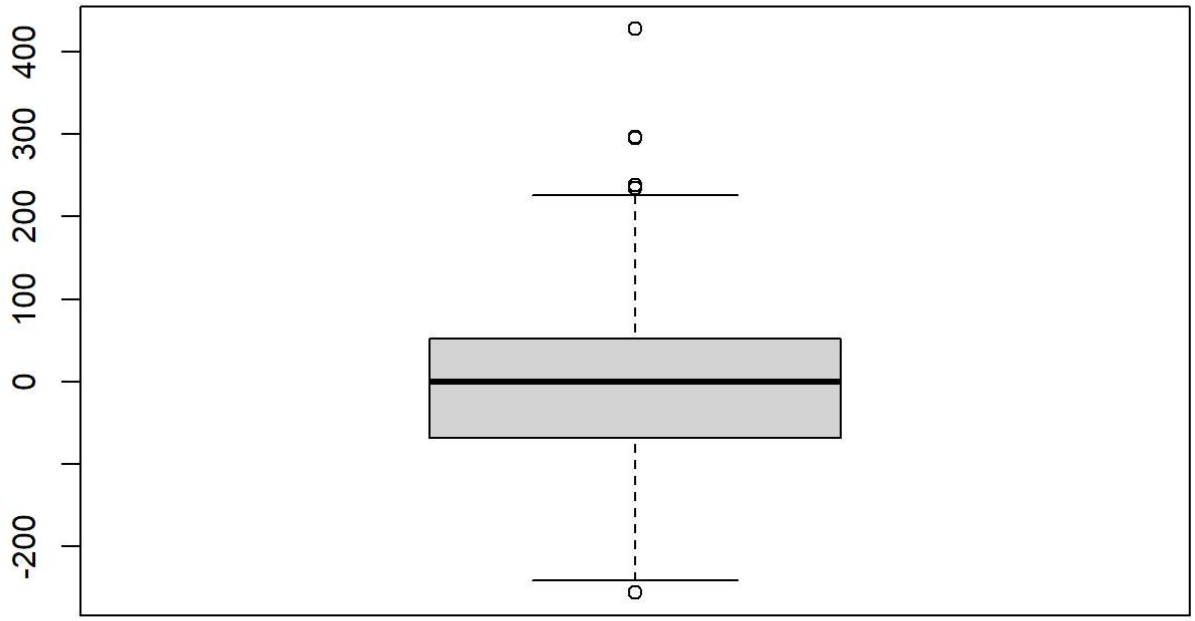

```
plot(f2$residuals)
```



```
summary(f2$residuals)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -255.5923  -67.4764    0.1357    0.0000   51.9849  427.0647
```

```
boxplot(f2$residuals)
```



#Caret

Caret actúa como interfaz única para otros muchos métodos. En total, hace de interfaz con 136 métodos de regresión distintos.

```
#install.packages("caret")
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
modelLookup()
```

	model <chr>	parameter <chr>
1	ada	iter
2	ada	maxdepth
3	ada	nu
4	AdaBag	mfinal
5	AdaBag	maxdepth
9	adaboost	nIter
10	adaboost	method
6	AdaBoost.M1	mfinal
7	AdaBoost.M1	maxdepth
8	AdaBoost.M1	coeflearn

1-10 of 502 rows | 1-3 of 7 columns Previous **1** 2 3 4 5 6 … 51 Next

```
trainIndex <- createDataPartition(dfR$BRAIN_VOLUME,
                                  p = .8,
                                  list = FALSE,
                                  times = 1)

b_train <- dfR[trainIndex, ]
```

```
## Warning: The `i` argument of ``[`()` can't be a matrix as of tibble 3.0.0.
## Convert to a vector.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

```
b_test  <- dfR[-trainIndex, ]
```

```
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 1)

# Y esta función entrena el método
caret_lm_model <- train(BRAIN_VOLUME ~ .,
                        data = b_train,
                        method = "lm",
                        trControl = fitControl)
```

```
caret_lm_model
```

```
## Linear Regression
##
## 210 samples
##   2 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 190, 190, 189, 189, 188, 190, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   105.2804  0.5099446  82.6416
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

Podemos observar que se ha conseguido un MSE = 11224.32.

Comparación entre modelos.

Con esto, somos capaces de seleccionar el mejor de los métodos (el que minimice el error cometido):

```
#install.packages("kableExtra", dependencies = TRUE)

# Recorremos todos los métodos y tomamos el RMSE de cada uno
best_rmse <- sapply(caret_results, function(i) min(i$results$RMSE, na.rm = TRUE))

# El mejor será el que tenga un RMSE más bajo
best_regressor <- names(best_rmse)[which.min(best_rmse)]

# Los mostramos en forma de tabla
w <- best_rmse %>% as.data.frame()

colnames(w) <- "RMSE"

w %>%
  knitr::kable(format = "html") %>% kableExtra::kable_styling(font_size = 14)
```

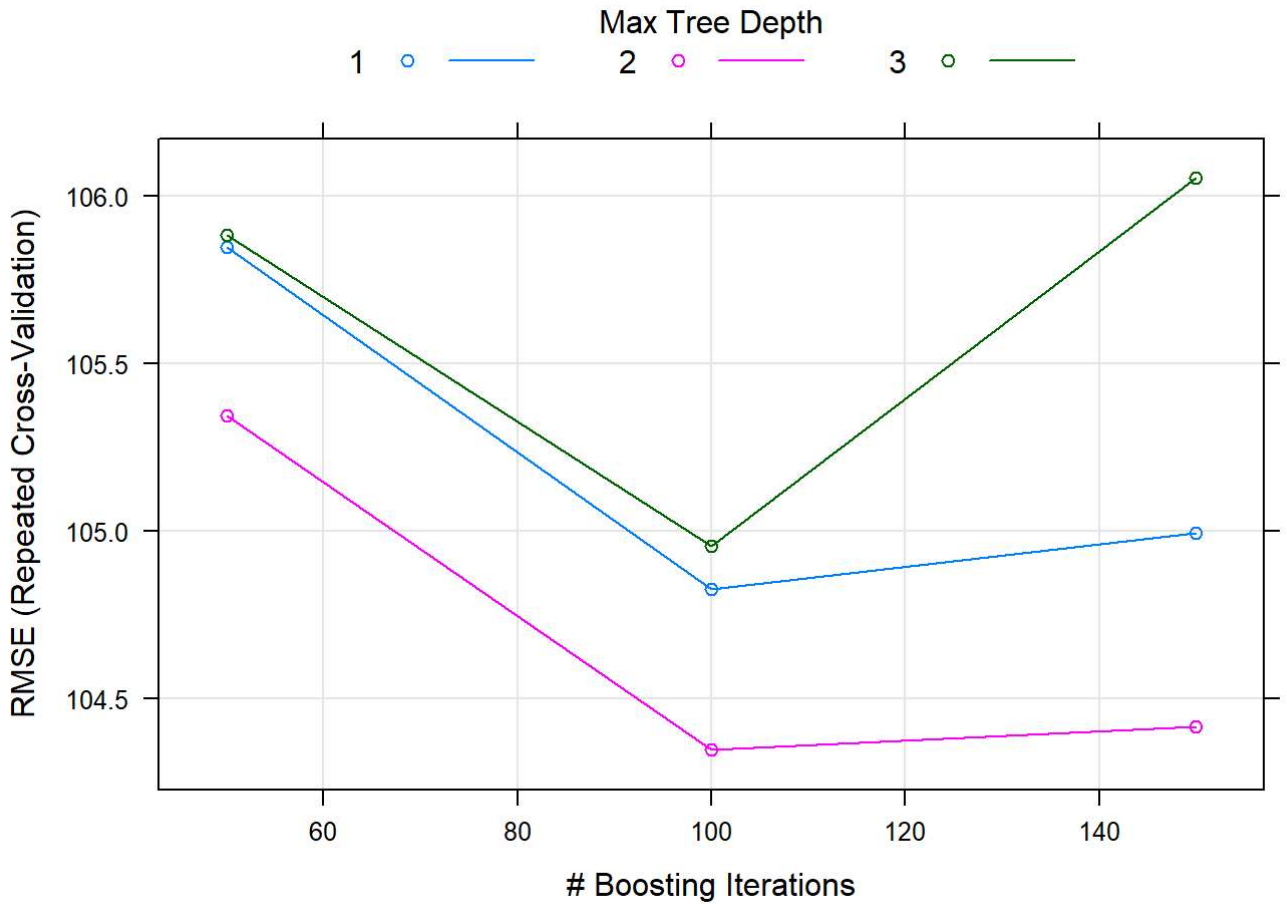
	RMSE
knn	118.6978
gbm	104.3459
lm	104.9709

```
cat("El metodo que minimiza la medida del error:",best_regressor)
```

```
## El metodo que minimiza la medida del error: gbm
```

Visualización de resultados.

```
plot(caret_results$gbm)
```



Vamos a repetir los 2 pasos previos para encontrar modelos que estimen la variable GM_VOLUME (volumen de sustancia gris) y modelos para estimar WM_VOLUME (volumen de sustancia blanca) a partir de la edad y del sexo.

```
dfRG= df2 %>% select(SEX,AGE,GM_VOLUME)

dfRW= df2 %>% select(SEX,AGE,WM_VOLUME)
```

Modelo de regresion simple y polinomico (volumen de sustancia gris).

```
modelo_1G <- lm(GM_VOLUME ~ AGE, data = dfRG)
modelo_2G <- lm(GM_VOLUME ~ poly(AGE, 2), data = dfRG)
modelo_3G <- lm(GM_VOLUME ~ poly(AGE, 3), data = dfRG)
modelo_4G <- lm(GM_VOLUME ~ poly(AGE, 4), data = dfRG)
modelo_5G <- lm(GM_VOLUME ~ poly(AGE, 5), data = dfRG)

anova(modelo_1G, modelo_2G, modelo_3G, modelo_4G, modelo_5G)
```

	Res.Df <dbl>	RSS <dbl>	Df <dbl>	Sum of Sq <dbl>	F <dbl>	Pr(>F) <dbl>
1	260	1111728	*NA*	*NA*	*NA*	*NA*
2	259	1068505	1	43222.7035	10.54771733	0.001318737
3	258	1068369	1	135.5686	0.03308306	0.855814967
4	257	1049153	1	19216.2270	4.68937189	0.031273500
5	256	1049043	1	109.8469	0.02680615	0.870076168

5 rows

```
f1G= lm(dfRG$GM_VOLUME ~ dfRG$AGE + dfRG$SEX)
f2G= lm(dfRG$GM_VOLUME ~ dfRG$AGE + dfRG$SEX + I(dfRG$AGE^2))
f3G= lm(dfRG$GM_VOLUME ~ dfRG$AGE + dfRG$SEX + I(dfRG$AGE^2) + I(dfRG$AGE^3))
f4G= lm(dfRG$GM_VOLUME ~ dfRG$AGE + dfRG$SEX + I(dfRG$AGE^2) + I(dfRG$AGE^3) + I(dfRG$AGE^4))
f5G= lm(dfRG$GM_VOLUME ~ dfRG$AGE + dfRG$SEX + I(dfRG$AGE^2) + I(dfRG$AGE^3) + I(dfRG$AGE^4)
 + I(dfRG$AGE^5))
anova(f1G,f2G,f3G,f4G,f5G)
```

	Res.Df <dbl>	RSS <dbl>	Df <dbl>	Sum of Sq <dbl>	F <dbl>	Pr(>F) <dbl>
1	259	837304.4	*NA*	*NA*	*NA*	*NA*
2	258	816401.5	1	20902.88414	6.724830804	0.01005734
3	257	812178.2	1	4223.28072	1.358704765	0.24485121
4	256	792634.0	1	19544.23669	6.287729676	0.01277996
5	255	792620.0	1	13.95796	0.004490525	0.94662516

5 rows

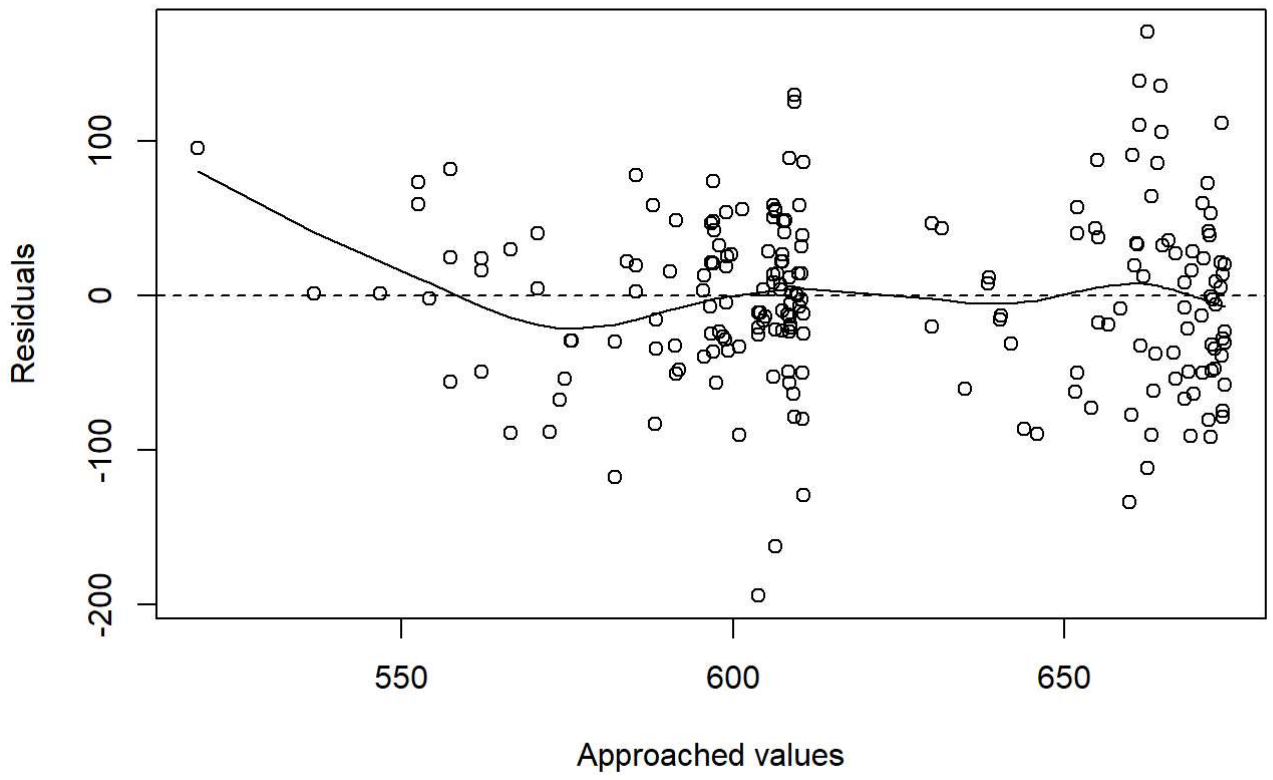
#degree 2 is the better

```
summary(f2G)
```

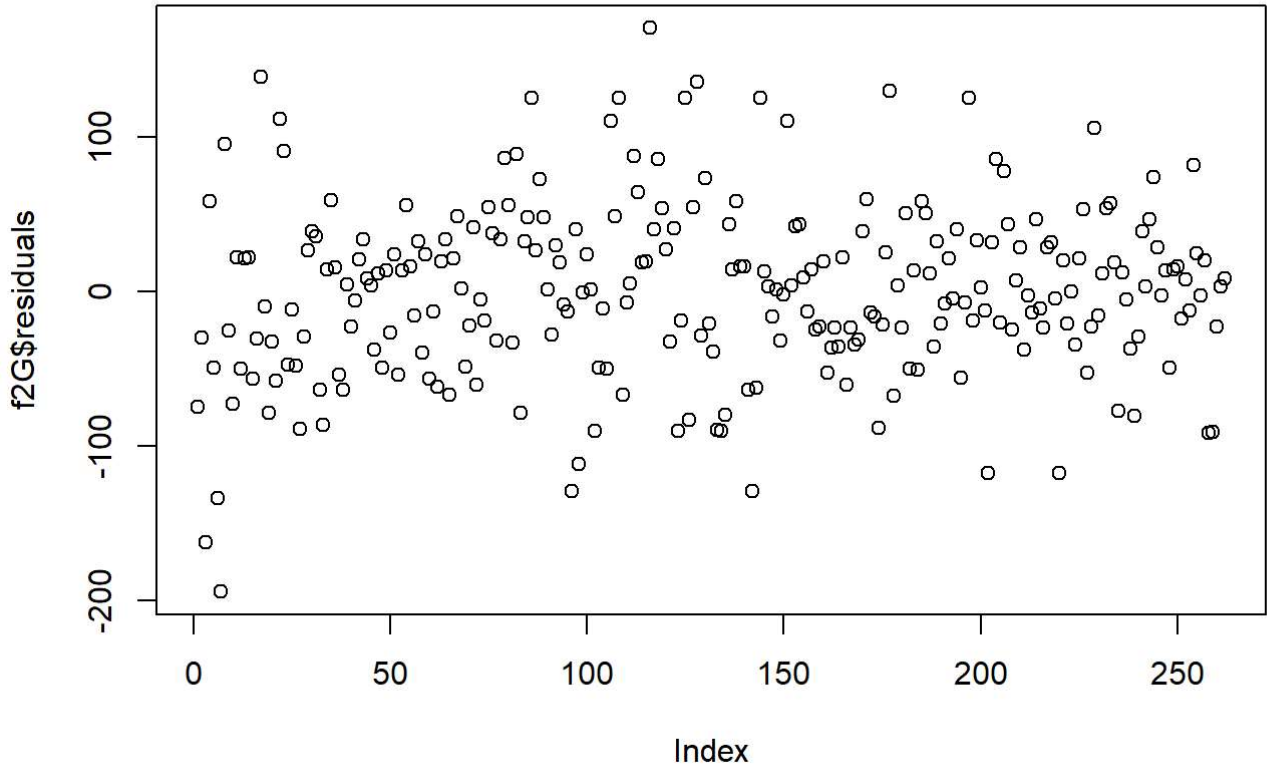
```
##
## Call:
## lm(formula = dfRG$GM_VOLUME ~ dfRG$AGE + dfRG$SEX + I(dfRG$AGE^2))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -194.297  -32.849    1.286   32.477  170.276
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   149.60912  223.41050   0.670   0.5037
## dfRG$AGE       13.88024    6.04425   2.296   0.0225 *
## dfRG$SEXMALE   63.50071    7.11428   8.926   <2e-16 ***
## I(dfRG$AGE^2)  -0.10447    0.04065  -2.570   0.0107 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 56.25 on 258 degrees of freedom
## Multiple R-squared:  0.2907, Adjusted R-squared:  0.2824
## F-statistic: 35.24 on 3 and 258 DF,  p-value: < 2.2e-16
```

p-value: < 2.2e-16 por lo tanto se acepta el modelo. En el caso del predictor AGE, si el resto de variables no varían, por cada unidad de AGE que aumenta el volumen cerebral se aumenta en promedio 13.88024 unidades. Las variables son significativas (al menos '*' 0.05), Coeficiente de determinacion R^2(0.2907 o 29%): es el porcentaje de la variación en la variable de respuesta que es explicado por un modelo lineal.

```
plot.new()
plot(fitted(f2G), residuals(f2G), xlab = "Approached values",  ylab = "Residuals")
  abline(h=0, lty=2)
    lines(smooth.spline(fitted(f2G), residuals(f2G)))
```



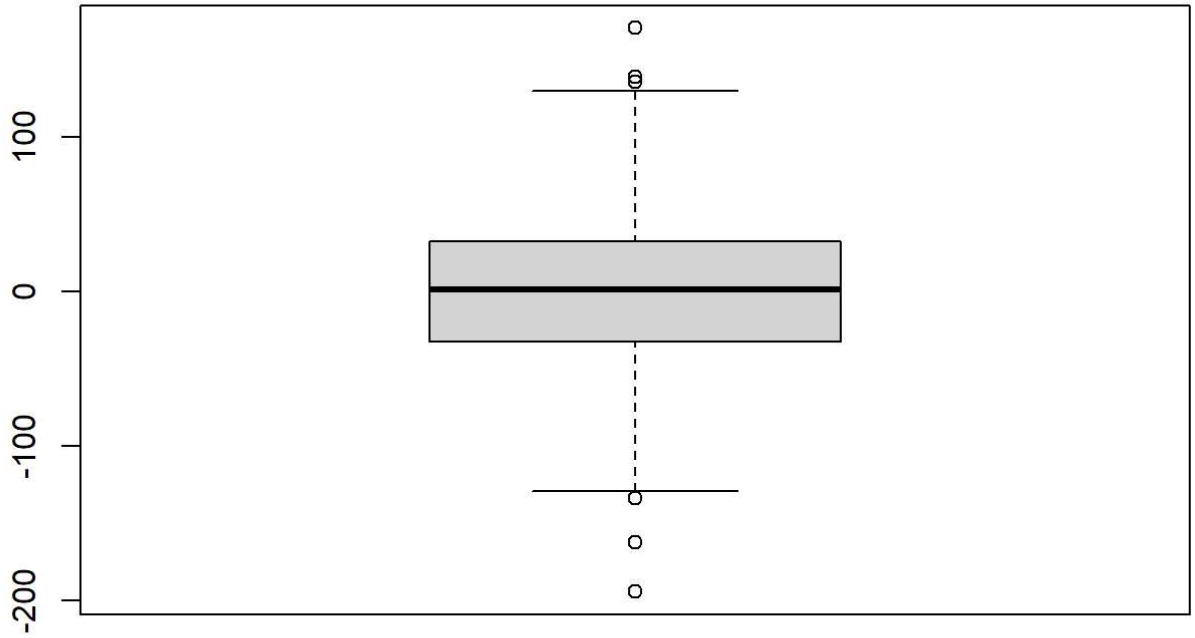

```
plot(f2G$residuals)
```




```
summary(f2G$residuals)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -194.297  -32.849    1.286    0.000   32.477  170.275
```

```
boxplot(f2G$residuals)
```



CARET para modelos del volumen de sustancia gris.

```
trainIndex <- createDataPartition(dfRG$GM_VOLUME,
                                  p = .8,
                                  list = FALSE,
                                  times = 1)

b_train <- dfRG[trainIndex, ]
b_test  <- dfRG[-trainIndex, ]
```

```
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 1)

# Y esta función entrena el método
caret_lm_model <- train(GM_VOLUME ~ .,
                        data = b_train,
                        method = "lm",
                        trControl = fitControl)
```

```
caret_lm_model
```

```
## Linear Regression
##
## 211 samples
##   2 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 189, 190, 190, 189, 191, 190, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   56.17812  0.2959678  43.71961
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

Podemos observar que se ha conseguido un MSE = 3155.691

Comparación entre modelos.

Con esto, somos capaces de seleccionar el mejor de los métodos (el que minimice el error cometido):

```
#install.packages("kableExtra", dependencies = TRUE)

# Recorremos todos los métodos y tomamos el RMSE de cada uno
best_rmse <- sapply(caret_results, function(i) min(i$results$RMSE, na.rm = TRUE))

# El mejor será el que tenga un RMSE más bajo
best_regressor <- names(best_rmse)[which.min(best_rmse)]

# Los mostramos en forma de tabla
w <- best_rmse %>% as.data.frame()

colnames(w) <- "RMSE"

w %>%
  knitr::kable(format = "html") %>% kableExtra::kable_styling(font_size = 14)
```

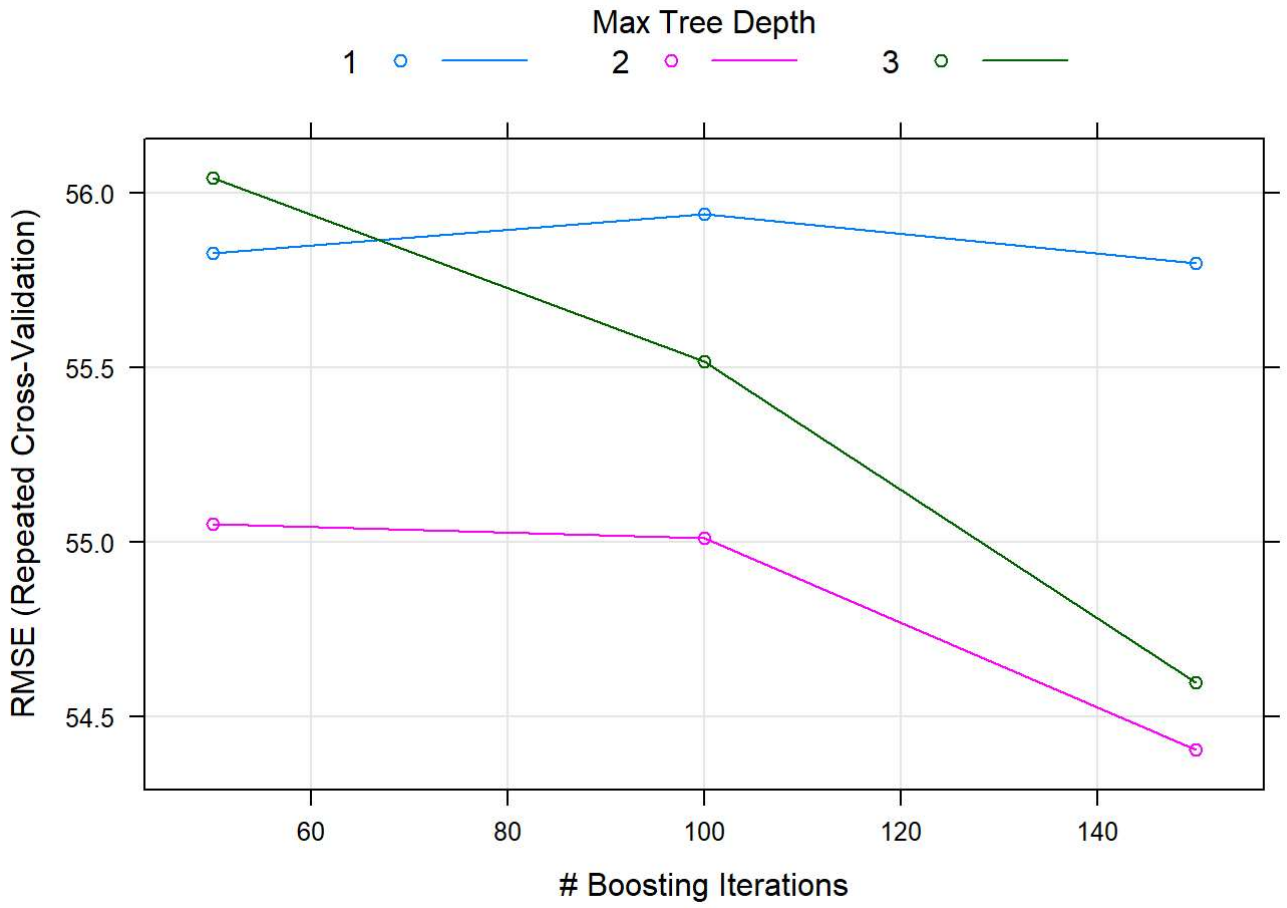
	RMSE
knn	57.85402
gbm	54.40479
lm	56.59584

```
cat("El metodo que minimiza la medida del error:",best_regressor)
```

```
## El metodo que minimiza la medida del error: gbm
```

Visualización de resultados.

```
plot(caret_results$gbm)
```



modelo de regresion simple y polinomico para modelos con volumen de sustancia blanca.

```
modelo_1W <- lm(WM_VOLUME ~ AGE, data = dfRW)
modelo_2W <- lm(WM_VOLUME ~ poly(AGE, 2), data = dfRW)
modelo_3W <- lm(WM_VOLUME ~ poly(AGE, 3), data = dfRW)
modelo_4W <- lm(WM_VOLUME ~ poly(AGE, 4), data = dfRW)
modelo_5W <- lm(WM_VOLUME ~ poly(AGE, 5), data = dfRW)

anova(modelo_1W, modelo_2W, modelo_3W, modelo_4W, modelo_5W)
```

	Res.Df <dbl>	RSS <dbl>	Df <dbl>	Sum of Sq <dbl>	F <dbl>	Pr(>F) <dbl>
1	260	801031.7	*NA*	*NA*	*NA*	*NA*
2	259	787861.6	1	13170.1685	4.3057267	0.03898323
3	258	787248.5	1	613.0759	0.2004331	0.65475012
4	257	783579.0	1	3669.4755	1.1996626	0.27441851
5	256	783041.6	1	537.4014	0.1756928	0.67545273

5 rows

```
f1W= lm(dfRW$WM_VOLUME ~ dfRW$AGE + dfRW$SEX)
f2W= lm(dfRW$WM_VOLUME ~ dfRW$AGE + dfRW$SEX + I(dfRW$AGE^2))
f3W= lm(dfRW$WM_VOLUME ~ dfRW$AGE + dfRW$SEX + I(dfRW$AGE^2) + I(dfRW$AGE^3))
f4W= lm(dfRW$WM_VOLUME ~ dfRW$AGE + dfRW$SEX + I(dfRW$AGE^2) + I(dfRW$AGE^3) + I(dfRW$AGE^4))
f5W= lm(dfRW$WM_VOLUME ~ dfRW$AGE + dfRW$SEX + I(dfRW$AGE^2) + I(dfRW$AGE^3) + I(dfRW$AGE^4)
 + I(dfRW$AGE^5))
anova(f1W,f2W,f3W,f4W,f5W)
```

	Res.Df <dbl>	RSS <dbl>	Df <dbl>	Sum of Sq <dbl>	F <dbl>	Pr(>F) <dbl>
1	259	474156.7	*NA*	*NA*	*NA*	*NA*
2	258	472150.2	1	2006.52049	1.09549649	0.2962476
3	257	470944.6	1	1205.63421	0.65823801	0.4179393
4	256	467114.8	1	3829.75048	2.09092219	0.1494045
5	255	467060.1	1	54.69547	0.02986199	0.8629408

5 rows

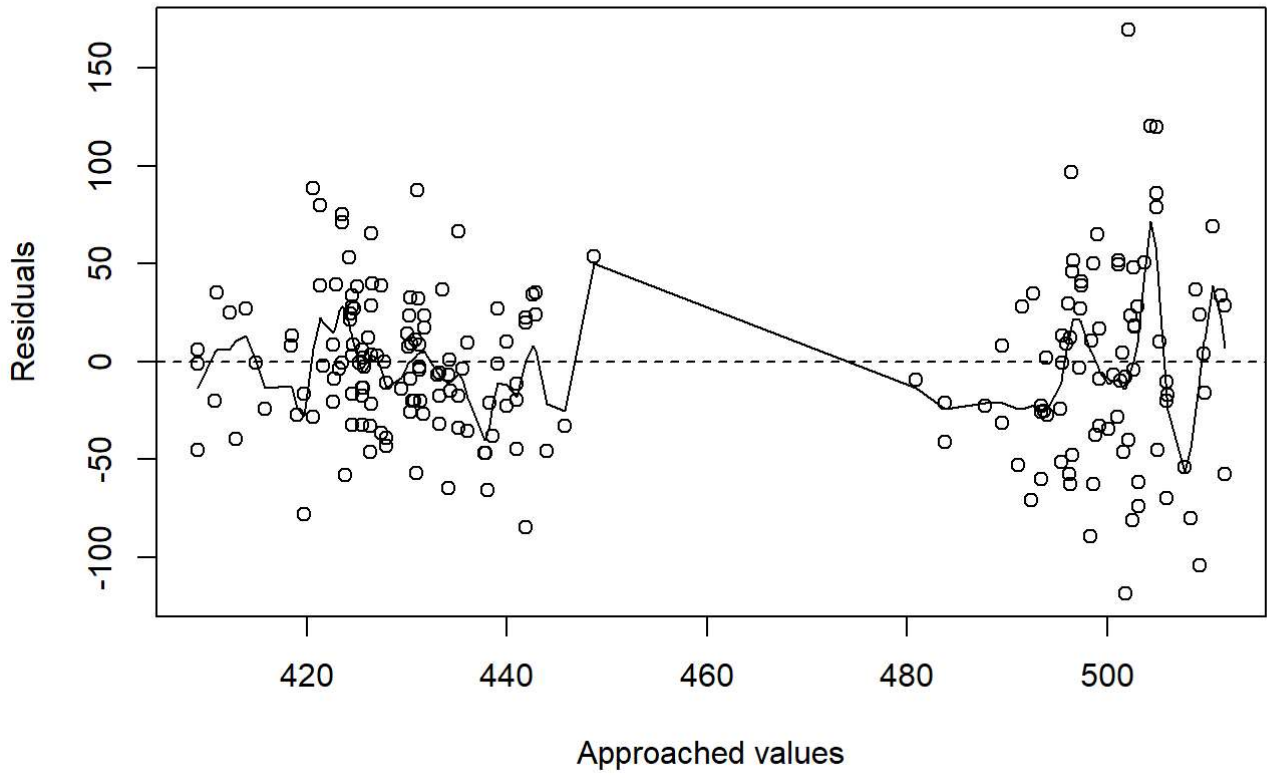
NINGUN MODELO CON P VALUE < 0.05, por lo que tomamos el modelo 1: dfRW $WM_VOLUME dfRW$AGE + dfRW$SEX

```
summary(f1W)
```

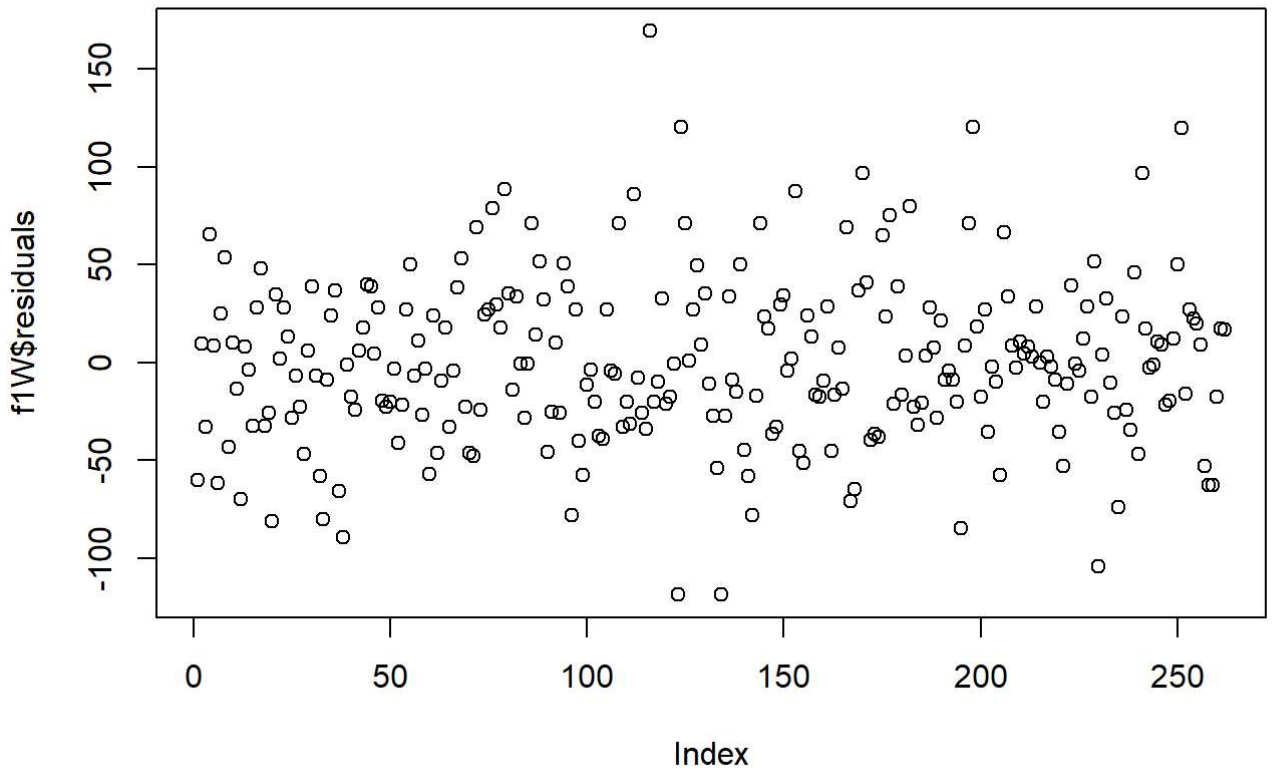
```
##
## Call:
## lm(formula = dfRW$WM_VOLUME ~ dfRW$AGE + dfRW$SEX)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -118.795  -25.695   -2.713   27.067  169.415
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  355.9619    26.4085  13.479  < 2e-16 ***
## dfRW$AGE       0.9657     0.3505   2.755  0.00629 **
## dfRW$SEXMALE  71.7581     5.3702  13.362  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 42.79 on 259 degrees of freedom
## Multiple R-squared:  0.4179, Adjusted R-squared:  0.4134
## F-statistic: 92.96 on 2 and 259 DF,  p-value: < 2.2e-16
```

p-value: < 2.2e-16 por lo tanto se acepta el modelo. En el caso del predictor AGE, si el resto de variables no varían, por cada unidad de AGE que aumenta el volumen cerebral se aumenta en promedio 0.9657 unidades. Las variables son significativas (al menos '*' 0.05), Coeficiente de determinacion R^2(0.4179 o 42%): es el porcentaje de la variación en la variable de respuesta que es explicado por un modelo lineal.

```
plot.new()
plot(fitted(f1W), residuals(f1W), xlab = "Approached values",  ylab = "Residuals")
  abline(h=0, lty=2)
    lines(smooth.spline(fitted(f1W), residuals(f1W)))
```



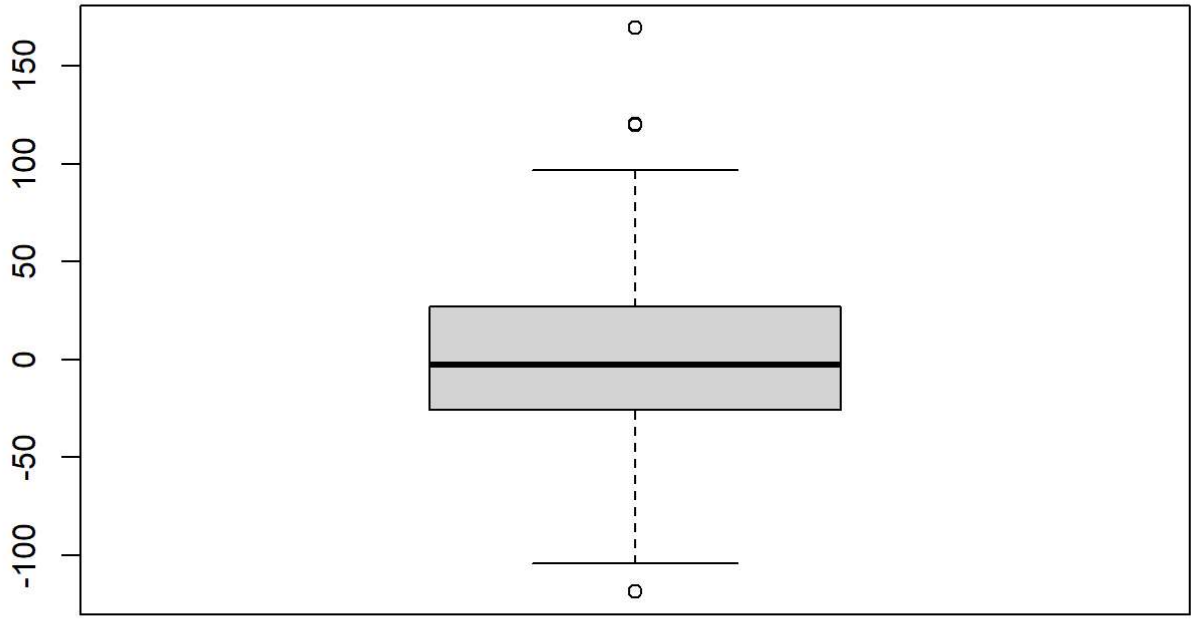

```
plot(f1W$residuals)
```




```
summary(f1W$residuals)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -118.795  -25.695   -2.713    0.000   27.067  169.415
```

```
boxplot(f1W$residuals)
```



CARET para modelos del volumen de sustancia blanca.

```
trainIndex <- createDataPartition(dfRW$WM_VOLUME,
                                  p = .8,
                                  list = FALSE,
                                  times = 1)

b_train <- dfRW[trainIndex, ]
b_test  <- dfRW[-trainIndex, ]
```

```
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 1)

# Y esta función entrena el método
caret_lm_model <- train(WM_VOLUME ~ .,
                        data = b_train,
                        method = "lm",
                        trControl = fitControl)
```

```
caret_lm_model
```

```
## Linear Regression
##
## 211 samples
##   2 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 190, 189, 190, 191, 190, 189, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   42.85845  0.4128418  33.93677
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

Podemos observar que se ha conseguido un MSE = 1825.792.

Comparación entre modelos.

```
# Métodos que queremos ejecutar
methods <- c("knn", "gbm", "lm")

# Los ejecutamos usando lapply
methods %>%
  lapply(function(x) {

    train(WM_VOLUME ~ .,
          data = b_train,
          method = x,
          trControl = fitControl,
          verbose = FALSE)

  }) -> caret_results
```

```
## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded
```

```
# A cada elemento del resultado le ponemos nombre
names(caret_results) <- methods
```

Con esto, somos capaces de seleccionar el mejor de los métodos (el que minimice el error cometido):

```
#install.packages("kableExtra", dependencies = TRUE)

# Recorremos todos los métodos y tomamos el RMSE de cada uno
best_rmse <- sapply(caret_results, function(i) min(i$results$RMSE, na.rm = TRUE))

# El mejor será el que tenga un RMSE más bajo
best_regressor <- names(best_rmse)[which.min(best_rmse)]

# Los mostramos en forma de tabla
w <- best_rmse %>% as.data.frame()

colnames(w) <- "RMSE"

w %>%
  knitr::kable(format = "html") %>% kableExtra::kable_styling(font_size = 14)
```

	RMSE

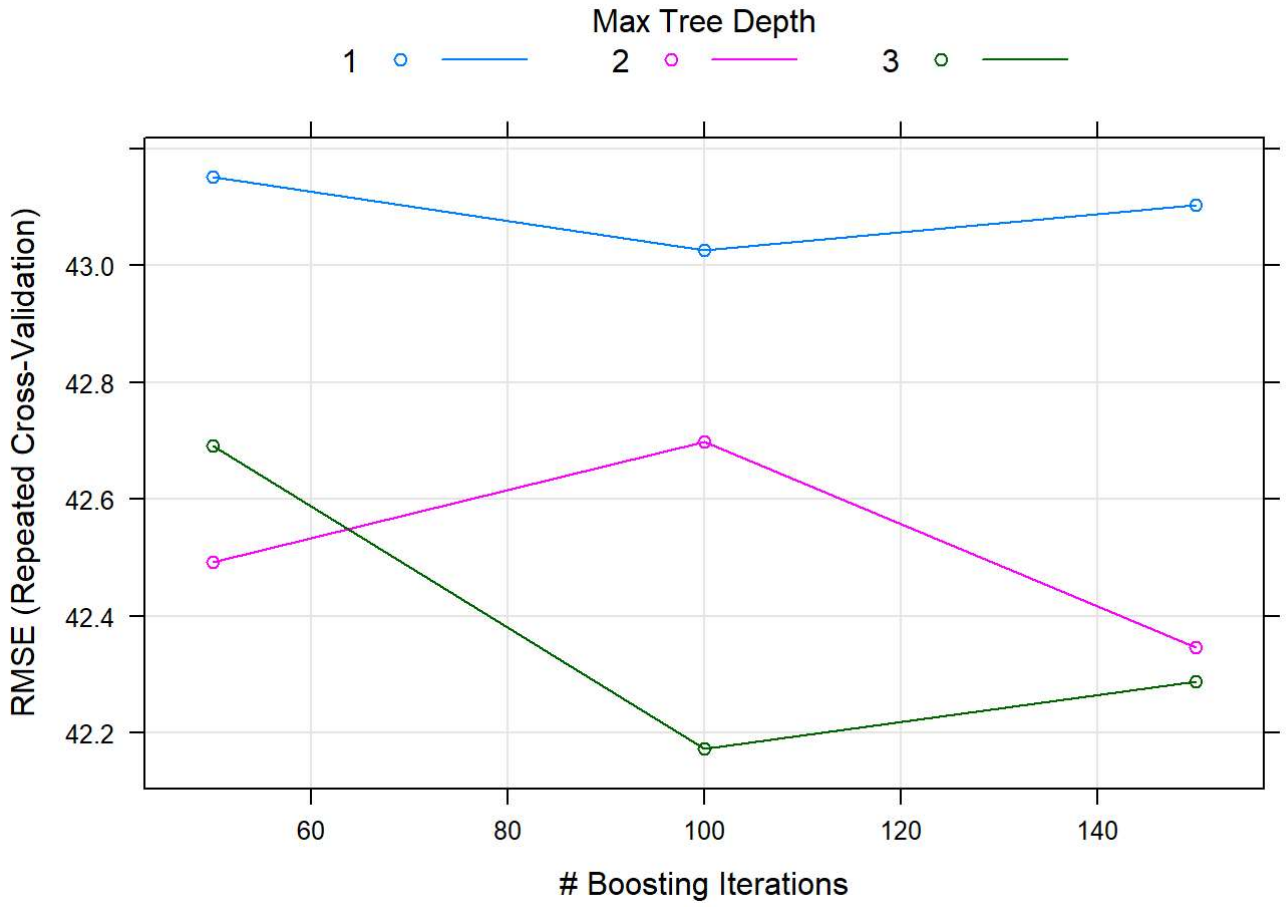
	RMSE
knn	48.33770
gbm	42.17258
lm	43.08988

```
cat("El metodo que minimiza la medida del error:",best_regressor)
```

```
## El metodo que minimiza la medida del error: gbm
```

Visualización de resultados.

```
plot(caret_results$gbm)
```



Usando como predictores las variables de volumen cebrebral de sustancia gris y blanca, así como el sexo, estimar la edad del individuo en cuestión utilizando caret.

```
dfRA= df2 %>% select(WM_VOLUME, GM_VOLUME,AGE,SEX)

trainIndex <- createDataPartition(dfRA$AGE,
                                  p = .8,
                                  list = FALSE,
                                  times = 1)

b_train <- dfRA[trainIndex, ]
b_test  <- dfRA[-trainIndex, ]
```

Comparación entre modelos.

```
# Métodos que queremos ejecutar
methods <- c("knn", "gbm", "lm")

# Los ejecutamos usando lapply
methods %>%
  lapply(function(x) {

    train(AGE ~ .,
          data = b_train,
          method = x,
          trControl = fitControl,
          verbose = FALSE)

  }) -> caret_results
```

```
## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded

## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'verbose' will be disregarded
```

```
# A cada elemento del resultado le ponemos nombre
names(caret_results) <- methods
```

Con esto, somos capaces de seleccionar el mejor de los métodos (el que minimice el error cometido):

```
#install.packages("kableExtra", dependencies = TRUE)

# Recorremos todos los métodos y tomamos el RMSE de cada uno
best_rmse <- sapply(caret_results, function(i) min(i$results$RMSE, na.rm = TRUE))

# El mejor será el que tenga un RMSE más bajo
best_regressor <- names(best_rmse)[which.min(best_rmse)]

# Los mostramos en forma de tabla
w <- best_rmse %>% as.data.frame()

colnames(w) <- "RMSE"

w %>%
  knitr::kable(format = "html") %>% kableExtra::kable_styling(font_size = 14)
```

	RMSE

	RMSE
knn	7.055963
gbm	6.698373
lm	6.789453

```
cat("El metodo que minimiza la medida del error:",best_regressor)
```

```
## El metodo que minimiza la medida del error: gbm
```

```
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 1)

# Y esta función entrena el método
caret_lm_model <- train(AGE ~ .,
                        data = b_train,
                        method = "lm",
                        trControl = fitControl)
```

Visualización de resultados.

```
caret_results$lm
```

```
## Linear Regression
##
## 211 samples
##   3 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 189, 189, 191, 191, 188, 191, ...
## Resampling results:
##
##   RMSE      Rsquared  MAE
##   6.789453  0.185895  5.344967
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

#Regresión logística y clasificación Con la regresión logística podemos asumir valores para variables binarias.

Queremos poder decidir, a partir de los datos de volumetría cerebral contemplados, si un nuevo sujeto puede padecer la enfermedad de Alzheimer.

Para ello, construiremos modelos de regresión logística y de clasificación para resolver ese problema.

Debemos dividir el dataset en un conjunto de entrenamiento con el 80% de los datos, seleccionado aleatoriamente, y un 20% restante para validar los modelos.

```
dfRL= df2 %>% select(BRAIN_VOLUME,WM_VOLUME, GM_VOLUME,AGE,SEX, CLASS)
set.seed(123)

dfRL$SEX = as.factor(dfRL$SEX)
dfRL$CLASS = as.factor(dfRL$CLASS)
trainIndex <- createDataPartition(dfRL$CLASS,
                                  p = .8,
                                  list = FALSE,
                                  times = 1)

b_train <- dfRL[trainIndex, ]
b_test  <- dfRL[-trainIndex, ]
```

```
gm <- glm(formula = CLASS ~ BRAIN_VOLUME  + WM_VOLUME + GM_VOLUME + AGE + SEX, data = b_trai
n, family = binomial)
summary(gm)
```

```
##
## Call:
## glm(formula = CLASS ~ BRAIN_VOLUME + WM_VOLUME + GM_VOLUME +
##     AGE + SEX, family = binomial, data = b_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7349   0.2763   0.3918   0.5279   1.1665
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.755961   3.378187  -0.224 0.822931
## BRAIN_VOLUME -0.022006   0.007108  -3.096 0.001963 **
## WM_VOLUME     0.027491   0.011415   2.408 0.016025 *
## GM_VOLUME     0.030768   0.008819   3.489 0.000485 ***
## AGE           0.031391   0.031152   1.008 0.313606
## SEXMALE       0.501730   0.660292   0.760 0.447337
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 149.26  on 209  degrees of freedom
## Residual deviance: 133.74  on 204  degrees of freedom
## AIC: 145.74
##
## Number of Fisher Scoring iterations: 5
```

EL modelo nos indica que los predictores volumen cerebral, sustancia blanca y gris son signiicativos, y que los predictores AGE y SEX pueden ser 0 por lo que serán omitidos para evitar ruido y mermar complejidad al modelo.

```
gm <- glm(formula = CLASS ~ BRAIN_VOLUME  + WM_VOLUME + GM_VOLUME, data = b_train, family = b
inomial)
summary(gm)
```

```
##
## Call:
## glm(formula = CLASS ~ BRAIN_VOLUME + WM_VOLUME + GM_VOLUME, family = binomial,
##     data = b_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7312   0.2638   0.3956   0.5485   1.0208
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.159136   2.110416   0.075 0.939892
## BRAIN_VOLUME -0.019525   0.006652  -2.935 0.003333 **
## WM_VOLUME     0.026763   0.011241   2.381 0.017270 *
## GM_VOLUME     0.028264   0.008479   3.333 0.000858 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 149.26  on 209  degrees of freedom
## Residual deviance: 135.16  on 206  degrees of freedom
## AIC: 143.16
##
## Number of Fisher Scoring iterations: 5
```

probar precison sobre el conjunto de validación

```
logit <- predict(gm,b_test)
p <- 1 / (1 + exp(-logit))
p
```

```
##         1         2         3         4         5         6         7         8
## 0.7274061 0.3787791 0.8467967 0.7318253 0.8275232 0.9781014 0.9130623 0.7390191
##         9        10        11        12        13        14        15        16
## 0.9578022 0.7478369 0.8323747 0.9792444 0.9512721 0.9492280 0.6901570 0.9586999
##        17        18        19        20        21        22        23        24
## 0.7333317 0.9461054 0.9688220 0.9343576 0.5164343 0.9003090 0.8785173 0.3836110
##        25        26        27        28        29        30        31        32
## 0.9688220 0.8738102 0.8436649 0.8979713 0.7377519 0.8670524 0.8955731 0.9476784
##        33        34        35        36        37        38        39        40
## 0.9032050 0.9714633 0.9536468 0.8756360 0.8016609 0.9625171 0.9489705 0.9609344
##        41        42        43        44        45        46        47        48
## 0.8655487 0.7240897 0.8580354 0.8395252 0.8381907 0.8878290 0.8852617 0.8090749
##        49        50        51        52
## 0.9837312 0.9294216 0.7714408 0.9331627
```

Crear un modelo de regresión logística adecuado (especialmente, en el que los predictores sean significativos) para la variable CLASS. ¿Qué precisión tiene ese modelo si lo ejecutamos (usando predict()) sobre el conjunto de validación?

Asimismo, se desea construir una serie de modelos de clasificación binaria con caret que, utilizando todos los parámetros disponibles, estime la clase (CLASS) a la que pertenece cada individuo. Para ello:

```
folds <- createFolds(dfRL$CLASS, k = 5)
# Validación cruzada
set.seed(123)
fitControl <- trainControl(method = "repeatedcv",
                           number = 5,
                           repeats = 1)
```

```
methods <- c("svmLinear", "rpart", "regLogistic", "C5.0")

# Se ejecutan
methods %>%
  lapply(function(x) {

    caret::train(CLASS ~ .,
                 data = b_train,
                 method = x,
                 trControl = fitControl)

  }) -> caret_class_results
```

```
## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials

## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials
```

```
## Warning: 'trials' should be <= 1 for this object. Predictions generated using 1
## trials
```

```
names(caret_class_results) <- methods
```

```
# Recorremos todos los resultados, almacenando la métrica
# accuracy (la precisión)
best_acc <- sapply(caret_class_results, function(i) max(i$results$Accuracy, na.rm = TRUE))

best_classifier <- names(best_acc)[which.max(best_acc)]

w <- best_acc %>% as.data.frame()

colnames(w) <- "Accuracy"

w %>%
  knitr::kable(format = "html") %>%
  kableExtra::kable_styling(font_size = 14)
```

	Accuracy
svmLinear	0.8858034
rpart	0.8858034
regLogistic	0.8905653
C5.0	0.8858034

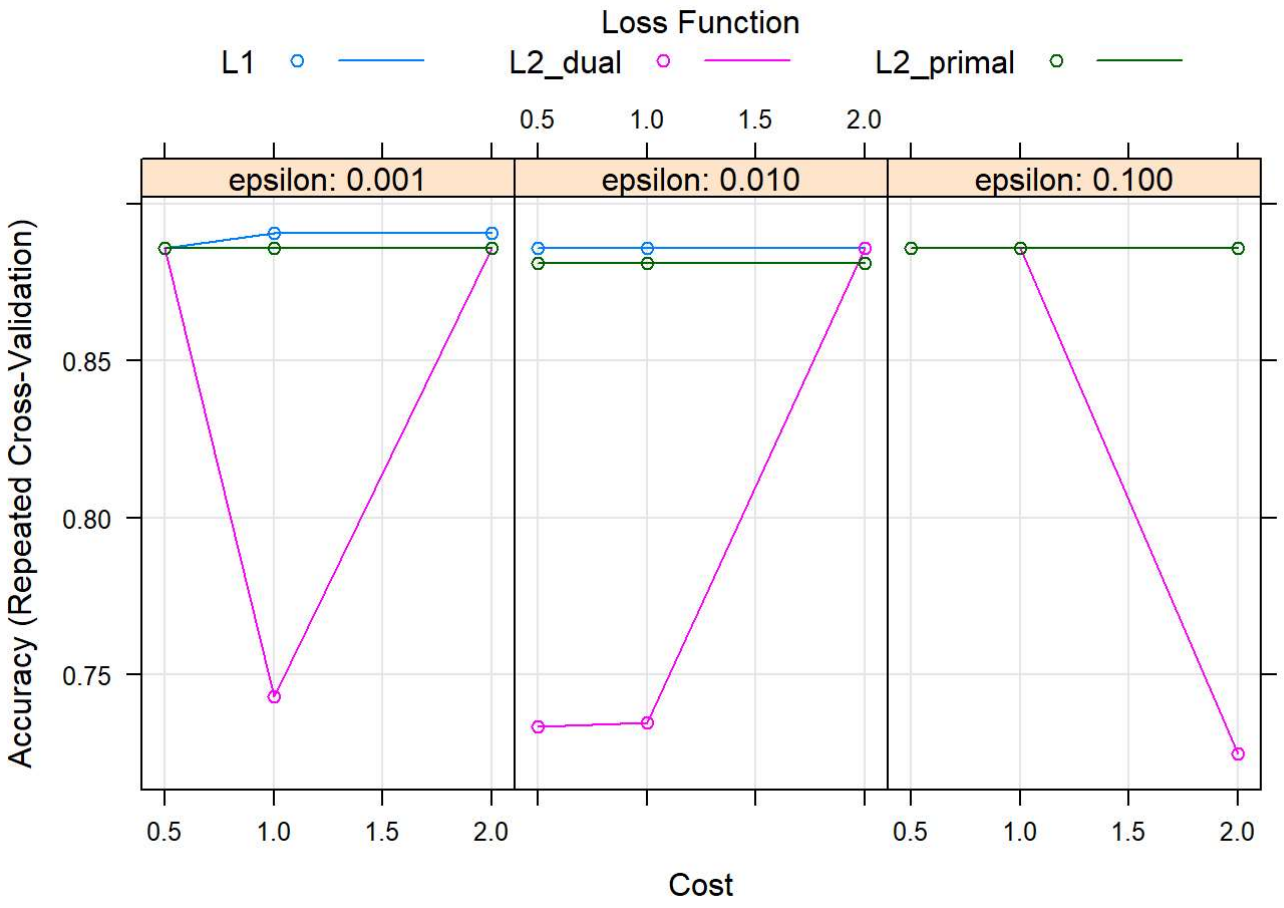
En este caso, el mejor clasificador ha sido regLogistic con una precisión de 89%.

```
predicted_class <- caret_class_results[[best_classifier]] %>%
  predict(b_test)

confusionMatrix(predicted_class, b_test$CLASS)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction AD HEALTHY
##    AD       1       1
##    HEALTHY  5      45
##
##                Accuracy : 0.8846
##                  95% CI : (0.7656, 0.9565)
##     No Information Rate : 0.8846
##     P-Value [Acc > NIR] : 0.6065
##
##                   Kappa : 0.2041
##
##  Mcnemar's Test P-Value : 0.2207
##
##             Sensitivity : 0.16667
##             Specificity : 0.97826
##          Pos Pred Value : 0.50000
##          Neg Pred Value : 0.90000
##              Prevalence : 0.11538
##          Detection Rate : 0.01923
##    Detection Prevalence : 0.03846
##       Balanced Accuracy : 0.57246
##
##        'Positive' Class : AD
##
```

```
plot(caret_class_results[[best_classifier]])
```



##Bibliografias

https://blog.minitab.com/es/analisis-de-regresion-como-puedo-interpretar-el-r-cuadrado-y-evaluar-la-bondad-de-ajuste (https://blog.minitab.com/es/analisis-de-regresion-como-puedo-interpretar-el-r-cuadrado-y-evaluar-la-bondad-de-ajuste)