Regresión múltiple. Datos de corte transversal.

Introducción.

En esta práctica vamos a explicar, mediante un modelo de regresión múltiple, el comportamiento de la rentabilidad económica (RENECO) de las empresas de producción eléctrica mediante tecnología eólica en función del grado de liquidez (LIQUIDEZ), del nivel de solvencia (SOLVENCIA), del margen aplicado (MARGEN), del grado de apalancamiento (APALANCA), y del tamaño del grupo corporativo al que pertenece (DIMENSION). Para ello se ha seleccionado una muestra constituida por 100 empresas.

Preparando Datos.

Abriremos R-Studio y crearemos nuestro **proyecto** siguiendo la instrucción File → New Project. Nos preguntará si crea el proyecto en una nueva carpeta o en una ya existente. Vamos a crearlo, por ejemplo, en el disco extraíble D, carpeta R, subcarpeta "regresion", que ya existe. Aparecerá una ventana para buscar la carpeta y, tras localizarla, pulsaremos Open y Create Project.

Vamos a ir a la carpeta del proyecto y vamos a guardar en ella los dos archivos de esta práctica: un archivo de Microsoft® Excel® llamado "eolica_100_mv_e.xlsx" y un *script* denominado "regresion_eolica.R". Si abrimos el archivo de Microsoft® Excel®, comprobaremos que se compone de tres hojas. La primera muestra el criterio de búsqueda de casos en la base de datos Sabi®; la segunda recoge la descripción de las variables consideradas, y la tercera (hoja "Datos") guarda los datos que debemos importar desde R-Studio. Estos datos se corresponden con diferentes

variables económico-financieras de 100 empresas productoras de electricidad mediante generación eólica.

Luego vamos a cerrar el archivo de Microsoft® Excel® y volveremos a R-Studio. Vamos a abrir nuestro script "regresion_eolica.R" con File → Open File… Este script contiene el programa que vamos a ir ejecutando en la práctica.

La primera línea / instrucción en el script es:

```
rm(list = ls())
```

La instrucción tiene como objeto limpiar el *Environment* de objetos de anteriores sesiones de trabajo. Para importar los datos, ejecutaremos el código:

```
#Importando
library(readxl)
eolicos <- read_excel("eolica_100_mv_e.xlsx", sheet = "Datos")
eolicos <- data.frame(eolicos, row.names = 1)
summary (eolicos)
```

Podemos observar cómo, en el *Environment,* ya aparece un objeto. Este objeto es una estructura de datos tipo *data frame*, se llama "eolicos" y contiene 13 columnas, una por cada variable del archivo de Microsoft® Excel®.

R ha considerado la primera columna (NOMBRE) como una variable de tipo cualitativo. En realidad, no es una variable, sino el nombre de los casos (empresas). Para evitar que R tome los nombres de los individuos como una variable, podemos redefinir nuestro *data frame* diciéndole que tome esa primera columna como los nombres de los individuos o casos (filas):

```
eolicos <- data.frame(eolicos, row.names = 1)
```

En la línea anterior hemos asignado al *data frame* "eolicos" los propios datos de "eolicos"; pero indicando que la primera columna de datos no es una variable; sino el nombre de los individuos, casos o filas.

Vemos que ya no aparece NOMBRE como variable, y en el *Environment* ya aparece el *data frame* "eolicos" con 100 observaciones (casos), pero con 12 variables (una menos).

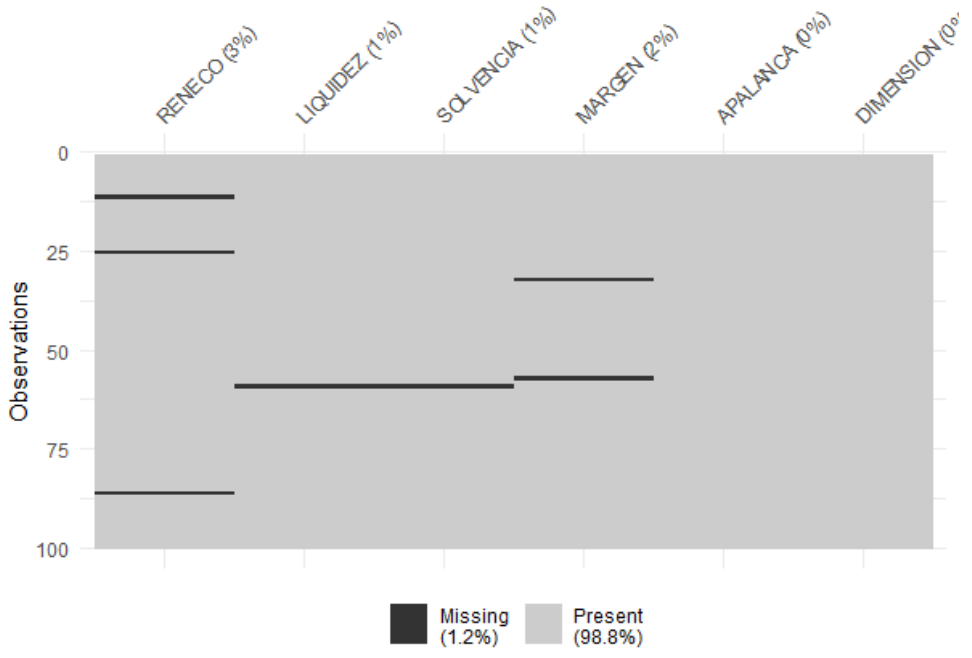
En nuestro análisis solo vamos a considerar las variables RENECO (variable dependiente), LIQUIDEZ, SOLVENCIA, MARGEN, APALANCA y DIMENSION. Por ello, crearemos con ellas un nuevo *data frame* llamado, por ejemplo, "originales":

```
#Seleccionando variables clasificadoras para el analisis

library(dplyr)
originales<-select(eolicos, RENECO, LIQUIDEZ, SOLVENCIA, MARGEN,
APALANCA, DIMENSION)
summary (originales)
```

El siguiente paso será localizar los posibles ***missing values***, ya que para realizar el análisis es necesario que todos los casos posean dato para todas las variables originales. Para tener una idea general, se puede utilizar la función vis_miss() del paquete visdat, que nos localizará gráficamente los *missing values* de las diferentes variables, y calculará el porcentaje de casos que supone, con respecto al total de observaciones:

```
# Identificando missing values.

library(visdat)
vis_miss(originales)
```

El resultado del código anterior es el siguiente gráfico:



Se detectan 7 missing values que afectan a 6 casos o empresas. Para localizarlos, podemos filtrar nuestro *data frame* con las herramientas de dplyr:

```
originales %>% filter(is.na(RENECO) | is.na(LIQUIDEZ) | is.na(SOLVENCIA) | is.na(MARGEN) | is.na(APALANCA) | is.na(DIMENSION)) %>%
  select(RENECO, LIQUIDEZ, SOLVENCIA, MARGEN, APALANCA, DIMENSION)
```

Los casos que poseen *missing values* son los siguientes:

	RENECO	LIQUIDEZ	SOLVENCIA	MARGEN	APALANCA	DIMENSION
Viesgo Renovables SL.	NA	0.272	65.883	11.818	13.330	MEDIO
Sargon Energias SLU	NA	0.188	-12.811	-615.625	-879.289	MEDIO
Parc Eolic Sant Antoni SL	1.361	0.839	13.964	NA	595.281	MEDIO
Eolica La Brujula SA	7.295	1.643	51.474	NA	85.954	GRANDE
Parques Eolicos San Lorenzo SL	13.806	NA	NA	36.810	9.119	PEQUEÑO
Brulles Eolica SL	NA	12.200	55.284	47.227	78.525	PEQUEÑO

Estos casos deberían ser analizados para estudiar la posibilidad de obtener los datos faltantes mediante la consulta de otras fuentes o algún proceso de estimación. Si no es posible, como supondremos en el ejemplo, y no son muchos, tendrán que ser eliminados, por ejemplo, con el código:

```
originales <- originales %>%
  filter(! is.na(RENECO) & ! is.na(LIQUIDEZ) & ! is.na(SOLVENCIA) & ! is.na(MARGEN) & ! is.na(APALANCA) & ! is.na(DIMENSION))
```

Como resultado, el *data frame* "originales" pasa a tener 94 casos.

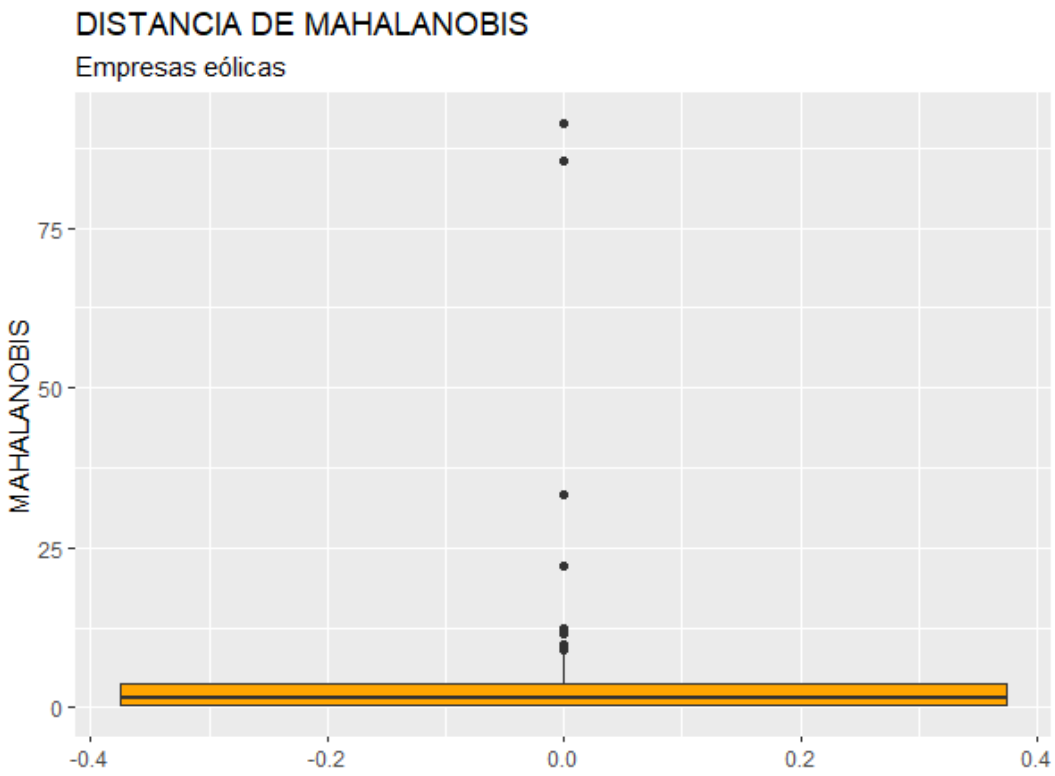
El siguiente paso será identificar los ***outliers*.** Para realizar esta tarea, y dado que en nuestro análisis contamos con 6 variables, primero "resumiremos" el valor que toman dichas variables para cada caso, mediante el cálculo de la *distancia de Mahalanobis*. No obstante, hay que tener en cuenta que la última variable del *data frame* "originales" (la sexta) es DIMENSION, que es un atributo, por lo que habrá que tener en cuenta que en el cálculo de la distancia **no se podrá incluir** (fijando el cálculo en las columnas 1 a 5). Las distancias de los diferentes casos se almacenarán en un vector, al que llamaremos MAHALANOBIS. Una vez calculado, lo añadiremos al *data frame* "originales_maha" mediante la función de pegado de columnas, cbind():

```
MAHALANOBIS <- mahalanobis(originales[,1:5],
                           center = colMeans(originales[,1:5]),
                           cov=cov(originales[,1:5]))
originales_maha <- cbind(originales, MAHALANOBIS)
```

A continuación, construiremos un diagrama de caja de la variable MAHALANOBIS a partir de la función ggplot() del paquete ggplot2:

```
library (ggplot2)
ggplot(data = originales_maha, map = (aes(y = MAHALANOBIS))) +
  geom_boxplot(fill = "orange") +
  ggtitle("DISTANCIA DE MAHALANOBIS", subtitle = "Empresas eólicas") +
  ylab("MAHALANOBIS")
```

El gráfico de caja resultante será:



En el gráfico se observa que existen, por encima de la caja, varios *outliers*. Para identificarlos de modo concreto, hemos de calcular los cuartiles primero y tercero de la variable MAHALANOBIS y pasar el correspondiente filtro:

```
Q1M <- quantile (originales_maha$MAHALANOBIS, c(0.25))
Q3M <- quantile (originales_maha$MAHALANOBIS, c(0.75))
originales_maha %>% filter(MAHALANOBIS > Q3M + 1.5*IQR(MAHALANOBIS) |
MAHALANOBIS < Q1M - 1.5*IQR(MAHALANOBIS)) %>% select(MAHALANOBIS)
```

Tras ejecutar el código anterior, se obtiene el siguiente listado de 13 casos que se comportan, según su *distancia de Mahalanobis* observada, como *outliers*:

```
                MAHALANOBIS
```

```
WPD Wind Investment SL.                 12.330454
Molinos Del Ebro SA                     33.267841
Parque Eolico Sierra De Las Carbas SL    9.113958
WPD Parque Eolico El Poleo SL.           9.973717
Tarraco Eolica SA                        9.553503
Elecdey Lezuza SA                       22.163771
WPD Parque Eolico Navillas SL.          85.439722
Eolica Navarra SL                       91.427674
Sierra De Selva SL                       9.890364
El Paramo Parque Eolico SL              12.401793
Parque Eolico El Moral SL               11.433290
```

Los outliers pueden originar distorsiones graves en los resultados de la regresión (cambio estructural, heteroscedasticidad, baja bondad del ajuste, etc.), por lo que conviene, si no son muchos, eliminarlos de la muestra. Para ello, puede recurrirse a ejecutar el siguiente código:

```
originales_so <- originales_maha %>% filter(MAHALANOBIS <= Q3M +
1.5*IQR(MAHALANOBIS) & MAHALANOBIS >= Q1M - 1.5*IQR(MAHALANOBIS))
originales_so <- originales_so %>% select(-MAHALANOBIS)
```

Con el código anterior, se ha creado un nuevo *data frame*, "originales_so", en el que ya no aparecen los *outliers*. Por tanto, puede observarse en el *Environment* cómo este *data frame* se compone de 83 casos. Por último, eliminaremos también del mismo la variable MAHALANOBIS, dejando solo las 6 variables originales.

Finalmente, es conveniente convertir la variable DIMENSION, atributo en escala ordinal, en una variable factor. Para ello, podrá ejecutarse:

```
#Convertir variable DIMENSION en Factor.

originales_so$DIMENSION <- as.factor(originales_so$DIMENSION)
levels(originales_so$DIMENSION)
```

Con lo que DIMENSION será considerada como un verdadero atributo o factor con los niveles o categorías:

```
[1] "GRANDE"  "MEDIO"   "PEQUEÑO"
```

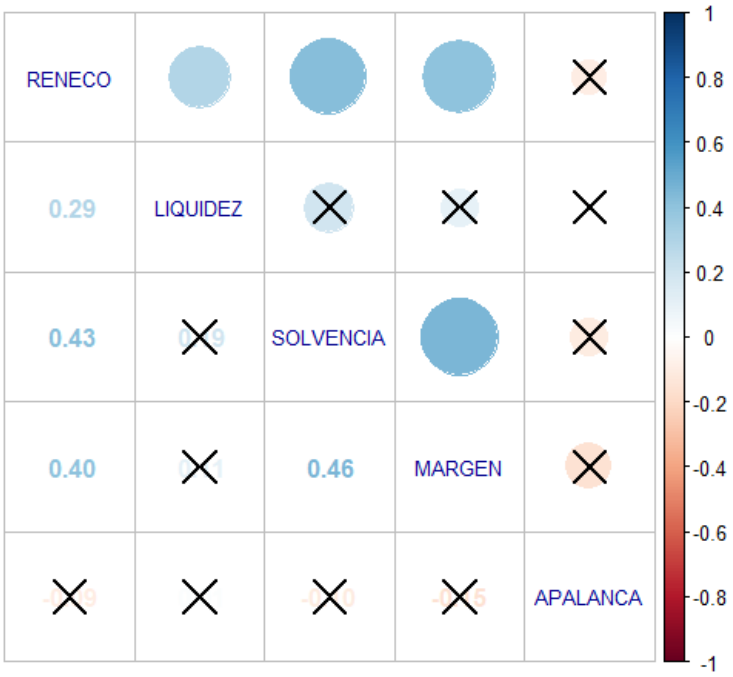
Especificación y estimación.

Algo que conviene hacer antes de especificar y estimar el modelo es analizar la **matriz de correlaciones** entre las variables en escala métrica. En general, es conveniente que las variables explicativas mantengan una **correlación alta (en términos absolutos) con la variable dependiente, y**

próxima a 0 entre ellas (para evitar problemas de multicolinealidad). Para calcular la matriz de correlaciones y presentarla de un modo visual atractivo, generaremos un nuevo *data frame* en el que se excluya al atributo o factor. A este *data frame* lo denominaremos, por ejemplo, "metricas". Después vamos a utilizar la función corrplot.mixed() del paquete corrplot. Entre los argumentos de la función encontramos "lower = "number"" para decir que en la parte inferior-izquierda aparezcan los valores de las correlaciones, "upper = "circle"" para fijar la parte superior-derecha con círculos de diámetro proporcional al valor absoluto de la correlación, "tl.cex =" para establecer el tamaño del texto y "tl.col =" para establecer el color del texto. Números y círculos tomarán tonos de color más sólidos en proporción a la intensidad de la correlación, en tonos azulados si la correlación es positiva, y en tonos tostados si es negativa. Finalmente, "p.mat = (cor.mtest(metricas)$p)" calcula, para una significación de 0,05, si las correlaciones son estadísticamente significativas o no, caso en el que añadirá en la celda del gráfico correspondiente un aspa:

```
#ESPECIFICACION Y ESTIMACION

#Correlaciones

metricas <- select(originales_so, -DIMENSION)
library (corrplot)
corrplot.mixed(cor(metricas), lower = "number", upper = "circle",
tl.cex = 0.8, tl.col = "dark blue", p.mat = (cor.mtest(metricas)$p))
```

El resultado del código anterior se concreta en la siguiente figura:



Se aprecia que hay correlaciones relativamente elevadas entre la variable dependiente (RENECO) con las variables explicativas LIQUIDEZ, SOLVENCIA y MARGEN; mientras que la correlación (en valor absoluto) con APALANCA es pobre y no significativa. Entre las explicativas, hay una correlación significativa entre SOLVENCIA y MARGEN, lo que podrá originar problemas de multicolinealidad en la estimación del modelo.

Para especificar y estimar el modelo inicial, deben de cargarse previamente algunos paquetes que es necesario utilizar, para lo cual se ejecutará el código:

```
## Proceso de especificacion

library (performance)      # comparación y diagnosis de modelos
library (gtsummary)        # presentación modelos
library (knitr)            # mejora presentación modelos
library (kableExtra)       # mejora presentación modelos
library (car)              # calcular vif
```

El modelo de regresión lineal, estimado mediante el método de mínimos cuadrados ordinarios (MCO), se obtendrá mediante la función lm(). Los resultados los guardaremos en un objeto de nombre, por ejemplo, "ecua0". Luego, se realizará un summary() para ver dicha estimación. Además, se pide una información adicional, el índice de inflación de la varianza vif(), que se obtiene a partir del paquete car. En definitiva, el código a ejecutar será:

```
ecua0 <- lm(data=originales_so, RENECO ~ LIQUIDEZ + SOLVENCIA + MARGEN
+ APALANCA + DIMENSION)
summary (ecua0)
vif (ecua0)
```

El resultado obtenido es:

```
Call:
lm(formula = RENECO ~ LIQUIDEZ + SOLVENCIA + MARGEN + APALANCA +
    DIMENSION, data = originales_so)

Residuals:
   Min     1Q Median     3Q    Max
-8.734 -1.675 -0.289  1.998  7.705

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       2.713e+00  8.509e-01   3.188  0.00208 **
LIQUIDEZ          8.101e-01  3.758e-01   2.156  0.03427 *
SOLVENCIA         3.130e-02  1.731e-02   1.808  0.07459 .
MARGEN            2.266e-02  9.209e-03   2.461  0.01614 *
APALANCA         -4.326e-05  2.453e-04  -0.176  0.86045
DIMENSIONMEDIO   -1.202e+00  9.243e-01  -1.301  0.19724
DIMENSIONPEQUEÑO -5.831e-01  1.346e+00  -0.433  0.66607
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.567 on 76 degrees of freedom
Multiple R-squared:  0.2981,  Adjusted R-squared:  0.2427
F-statistic:  5.38 on 6 and 76 DF,  p-value: 0.0001124

              GVIF Df GVIF^(1/(2*Df))
LIQUIDEZ  1.109125  1        1.053150
SOLVENCIA 1.496610  1        1.223360
MARGEN    1.309138  1        1.144176
APALANCA  1.037426  1        1.018541
DIMENSION 1.250107  2        1.057394
```

Antes de comentar los resultados, presentaremos estos de un modo más atractivo. Para ello, podemos exportar los resultados en modo tabla “html” personalizada, mediante la función tbl_regression() del paquete gtsummary. Este paquete permite presentar resultados de la estimación de modelos, y tablas, en un modo personalizado, usando una gramática parecida al paquete kableExtra. Así, crearemos el objeto de nombre, por ejemplo, “tbl_equa0”, que tomará el modelo “ecua0” estimado y lo presentará según las especificaciones que hagamos con tbl_regression() y otras funciones y paquetes:

```
knitr.table.format = "html"
tbl_equa0 <- ecua0 %>%
  tbl_regression(intercept = TRUE,
                 conf.int = FALSE,
                 estimate_fun = ~style_number(.x, digits = 4),
```

```
                  pvalue_fun = ~style_pvalue(.x, digits = 3)) %>%
  add_global_p() %>%
  bold_p() %>%
  add_vif() %>%
  add_glance_table(label = list(sigma ~ "\U03C3"),
                   include = c(nobs, sigma, r.squared, adj.r.squared,
AIC, statistic, p.value)) %>%
  modify_caption(caption = "Regresión Inicial")
```

Dentro de la función tbl_regression(), el argumento “intercept =” se refiere a si se desea que aparezca un término independiente , “conf.int” controla si se quieren mostrar los intervalos de confianza de los parámetros estimados, “estimate_fun =” y “pvalue_fun =” personalizan, respectivamente, el modo en el que se muestran los parámetros estimados y los p-valores correspondientes al contraste de significación individual de los parámetros. Posteriormente, se añade la función add_global_p() para que se muestre, caso que haya un atributo o factor entre las variables explicativas, un p-valor global para la significación de todas las categorías del atributo o factor. La función bold_p() provoca que los p-valores correspondientes a los parámetros estadísticamente significativos para el 0,05 de significación aparezcan en negrita. La función add_vif() añade columnas para mostrar los valores del *factor de inflación de la varianza (vif)*, y su versión ajustada. La función add_glance_table() sirve para añadir otras informaciones o resultados, que se insertan en el argumento “include =”. Finalmente, la función modify_caption() sirve para poner un título a la tabla con los resultados del modelo.

Aún se puede personalizar más la tabla, transformando el objeto “tbl_equa” en un objeto interpretable por el paquete tableExtra, y utilizando las funciones de este paquete para añadir otras especificaciones estéticas. La función para realizar esta transformación es as_kable_extra():

```
tbl_equa0 <- as_kable_extra(tbl_equa0,
                            booktabs = TRUE,
                            longtable = TRUE,
                            linesep = "") %>%
  kable_styling(full_width = F, bootstrap_options = "striped",
"bordered", "condensed", position = "center", font_size = 12)
tbl_equa0
```

El resultado obtenido, finalmente, es el siguiente:

Regresión Inicial

Characteristic	Beta	p-value	GVIF	Adjusted GVIF
(Intercept)	2.7128	**0.002**		
LIQUIDEZ	0.8101	**0.034**	1.1	1.1
SOLVENCIA	0.0313	0.075	1.5	1.2
MARGEN	0.0227	**0.016**	1.3	1.1
APALANCA	0.0000	0.860	1.0	1.0
DIMENSION		0.431	1.3	1.1
GRANDE	—			
MEDIO	-1.2024			
PEQUEÑO	-0.5831			
No. Obs.	83			
σ	3.57			
R^2	0.298			
Adjusted R^2	0.243			
AIC	455			
Statistic	5.38			
p-value	<0.001			

[1] GVIF = Generalized Variance Inflation Factor

[2] GVIF^[1/(2*df)]

Vemos que la estimación de esta especificación inicial ofrece un R^2 corregido de 0.243 (muy bajo). Los parámetros estimados son estadísticamente significativos en su conjunto (prueba de la F con p-valor menor que 0.05). Considerados individualmente, los parámetros asociados a las variables APALANCA, DIMENSION y SOLVENCIA no son estadísticamente significativos con una significación de 0.05 (contrastes t con p-valores mayores a 0.05). En cambio, los parámetros estimados asociados a LIQUIDEZ y MARGEN sí son estadísticamente significativos.

El factor o atributo DIMENSION se incluye en la especificación mediante tantas variables dicotómicas o "dummies" como niveles o categorías existen, menos uno. Deben ser interpretados en relación con el nivel o categoría que falta (que suele ser el primero, por orden alfabético). Por ejemplo, aquí el nivel que falta es "GRANDE". Así, el parámetro asociado al nivel "MEDIO" se interpreta como que, por el hecho de que una empresa pertenezca a un grupo empresarial de dimensión mediana, su rentabilidad económica disminuirá en 1,20 puntos con respecto a si perteneciera a un grupo empresarial "GRANDE".

Por último, es muy importante, a fin de evitar posibles problemas de multicolinealidad, valorar el *factor de inflación de la varianza*. Variables con un *vif* alto, superior a un valor de 5 (según algunos autores), deben ser consideradas como candidatas a desaparecer de la especificación de la regresión. En este ejemplo, todas las variables tienen un *vif corregido* inferior a 5, por lo que, tal y como se presumió al analizar el correlograma, es asumible pensar que ninguna de ellas origina un problema de multicolinealidad.

Existen métodos automatizados para que, una vez se tiene la estimación inicial realizada, se obtenga una especificación más sencilla (Principio de Parsimonia) sin una pérdida grande de validez de la regresión. Por ejemplo, un método es el *step / backward*, que, en función del *Criterio de Información de Akaike (AIC)*, irá probando a estimar especificaciones más simples que disminuyan de AIC (lo que implica una mejor especificación). En nuestro caso, se aplicará con el código:

```
ecuaDEF <- step(ecua0, scale = 0,
                  direction = c("backward"),
                  trace = 1, steps = 1000, k = 2)
summary (ecuaDEF)
vif (ecuaDEF)
```

La regresión estimada conforme a la especificación mejor, según el método *step / backward,* se ha guardado en el objeto "ecuaDEF". Después, se visualiza la solución y se añade el valor del *vif* de las variables:

```
Call:
lm(formula = RENECO ~ LIQUIDEZ + SOLVENCIA + MARGEN, data =
originales_so)

Residuals:
```

```
    Min      1Q  Median      3Q     Max
-8.5764 -1.9389 -0.3428  2.1611  8.0917

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 2.053810   0.688009   2.985  0.00377 **
LIQUIDEZ    0.801367   0.360905   2.220  0.02925 *
SOLVENCIA   0.039621   0.016021   2.473  0.01554 *
MARGEN      0.021405   0.008993   2.380  0.01971 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.54 on 79 degrees of freedom
Multiple R-squared:  0.2816,  Adjusted R-squared:  0.2543
F-statistic: 10.32 on 3 and 79 DF,  p-value: 8.319e-06

> vif (ecuaDEF)
 LIQUIDEZ SOLVENCIA    MARGEN
 1.038865  1.301280  1.267753
```

Para tener una presentación más elegante de los resultados, se puede realizar el mismo procedimiento que se hizo con la regresión inicial y la función la tbl_regression() del paquete gtsummary, esta vez creando un objeto de nombre, por ejemplo, “tbl_equaDEF”:

```
## Modelo final: presentacion.

tbl_equaDEF <- ecuaDEF %>%
  tbl_regression(intercept = TRUE,
                 conf.int = FALSE,
                 estimate_fun = ~style_number(.x, digits = 4),
                 pvalue_fun = ~style_pvalue(.x, digits = 3)) %>%
  add_global_p() %>%
  bold_p() %>%
  add_vif() %>%
  add_glance_table(label = list(sigma ~ "\U03C3"),
                   include = c(nobs, sigma, r.squared, adj.r.squared,
AIC, statistic, p.value)) %>%
  modify_caption(caption = "Regresión Final")

tbl_equaDEF <- as_kable_extra(tbl_equaDEF,
                              booktabs = TRUE,
                              longtable = TRUE,
                              linesep = "") %>%
  kable_styling(full_width = F, bootstrap_options = "striped",
"bordered", "condensed", position = "center", font_size = 12)
tbl_equaDEF
```

La presentación obtenida es:

Regresión Final

Characteristic	Beta	p-value	VIF
(Intercept)	2.0538	**0.004**	
LIQUIDEZ	0.8014	**0.029**	1.0
SOLVENCIA	0.0396	**0.016**	1.3
MARGEN	0.0214	**0.020**	1.3
No. Obs.	83		
σ	3.54		
R^2	0.282		
Adjusted R^2	0.254		
AIC	451		
Statistic	10.3		
p-value	<0.001		

[1] VIF = Variance Inflation Factor

El criterio de información de Akaike (AIC) ha disminuido, lo que significa que esta segunda especificación es mejor. Así lo corrobora también el R^2 corregido. Los parámetros estimados son significativos en su conjunto (prueba F con un p-valor menor que 0.05), y los parámetros asociados a todas las variables son estadísticamente significativos (p-valor menor que 0.05). Los valores del *vif* son muy reducidos, luego no existe peligro de que haya un problema de multicolinealidad entre las variables explicativas.

El paquete performance, mediante la función compare_performance(), permite comparar las regresiones inicial y final en cuanto a una batería de indicadores de bondad. El resultado de la función puede pasarse fácilmente a una tabla personalizada con extraKable:

```
tablacompara <- compare_performance(ecua0, ecuaDEF, rank = TRUE)
tablacompara %>%
  kable(caption = "Comparación de modelos Rentabilidad Económica") %>%
```

```
  kable_styling(full_width = F, bootstrap_options = "striped",
"bordered", "condensed", position = "center", font_size = 12) %>%
  row_spec(0, bold= T, align = "c")
```

El resultado es el siguiente:

Comparación de modelos Rentabilidad Económica

Name	Model	R2	R2_adjusted	RMSE	Sigma	AIC_wt	BIC_wt	Performance_Score
ecuaDEF	lm	0.2815633	0.2542809	3.453439	3.539788	0.8840883	0.9965295	0.6666667
ecua0	lm	0.2981317	0.2427210	3.413385	3.567119	0.1159117	0.0034705	0.3333333

El indicador "Performance_Score" es una síntesis de las medidas de bondad, e indica claramente cómo la regresión final, "ecuaDEF", es mejor que la inicial en términos de validez (representación de la realidad).

Contrastación de hipótesis básicas del modelo.

El siguiente paso, una vez se ha seleccionado una especificación, es realizar las pruebas o diagnosis del modelo que nos informen de si existen puntos "débiles" en cuanto a la verificación de las hipótesis básicas del modelo lineal y, en su caso, proponer algún método de mejora (lo que podría llevar incluso a volver a proponer una nueva especificación, estimar la regresión mediante un método más complejo a mínimos cuadrados ordinarios, o simplemente a "convivir" con el "problema").

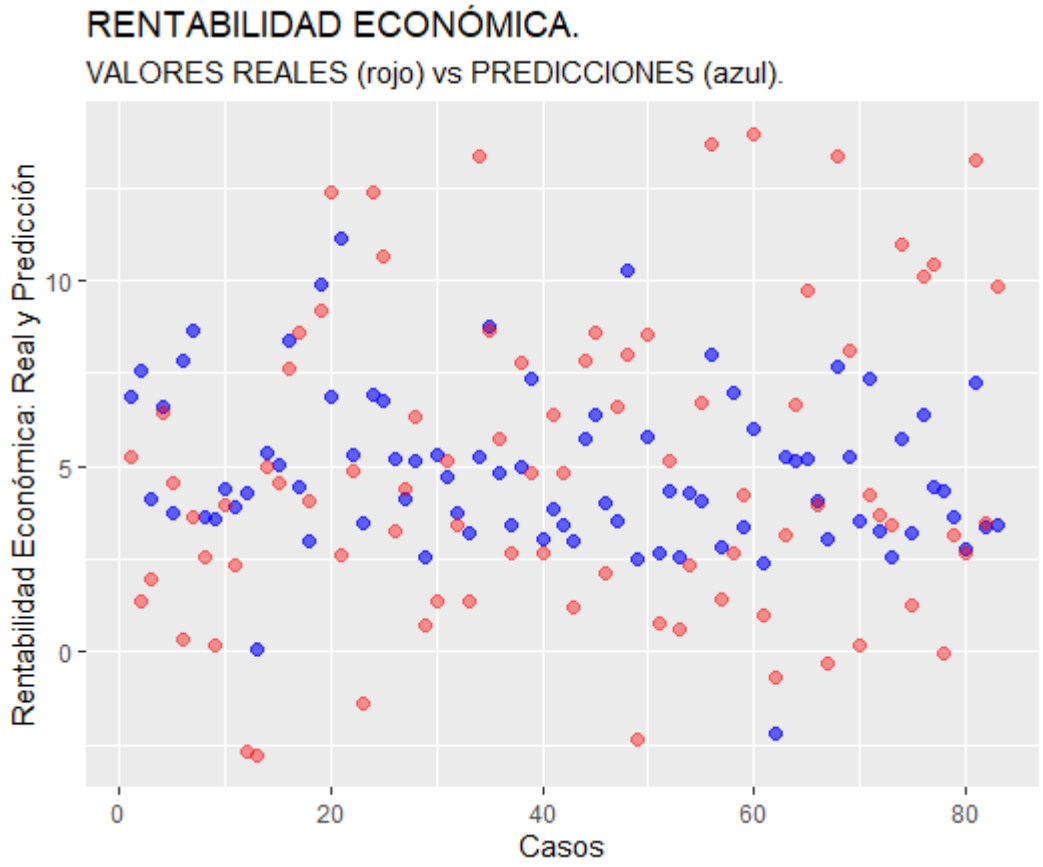
Es conveniente, previamente al análisis de cada hipótesis, generar dos gráficos de gran utilidad. Para ello, crearemos el *data frame,* "resultados", que contiene las predicciones para la muestra de la variable dependiente (elemento "fitted.values" del objeto "ecuaDEF"), los residuos (elemento "resid" del objeto "ecuaDEF"), además de los valores reales de la variable RENECO y una variable ORDEN que es un simple contador que toma valores desde 1 al número de casos del *data frame* "originales_so":

```
## Modelo final: Contrastación.

numcasos <- nrow(originales_so)
resultados          <-          data.frame(predic=ecuaDEF$fitted.values,
residuos=ecuaDEF$residuals,    RENECO=originales_so$RENECO,    ORDEN    =
c(1:numcasos))
```

El siguiente gráfico nos proporciona las predicciones que el modelo estimado hace sobre la variable dependiente (RENECO) para los casos de la muestra. Se utiliza la librería ggplot2 y su gramática:

```
ggplot(data = resultados)+
  geom_point(aes(x= ORDEN, y=predic), size= 2, alpha= 0.6, color =
"blue") +
  geom_point(aes(x= ORDEN, y=RENECO), size= 2, alpha= 0.4, color =
"red") +
  ggtitle("RENTABILIDAD ECONÓMICA.", subtitle= "VALORES REALES (rojo)
vs PREDICCIONES (azul).")+
  xlab("Casos") +
  ylab("Rentabilidad Económica: Real y Predicción")
```

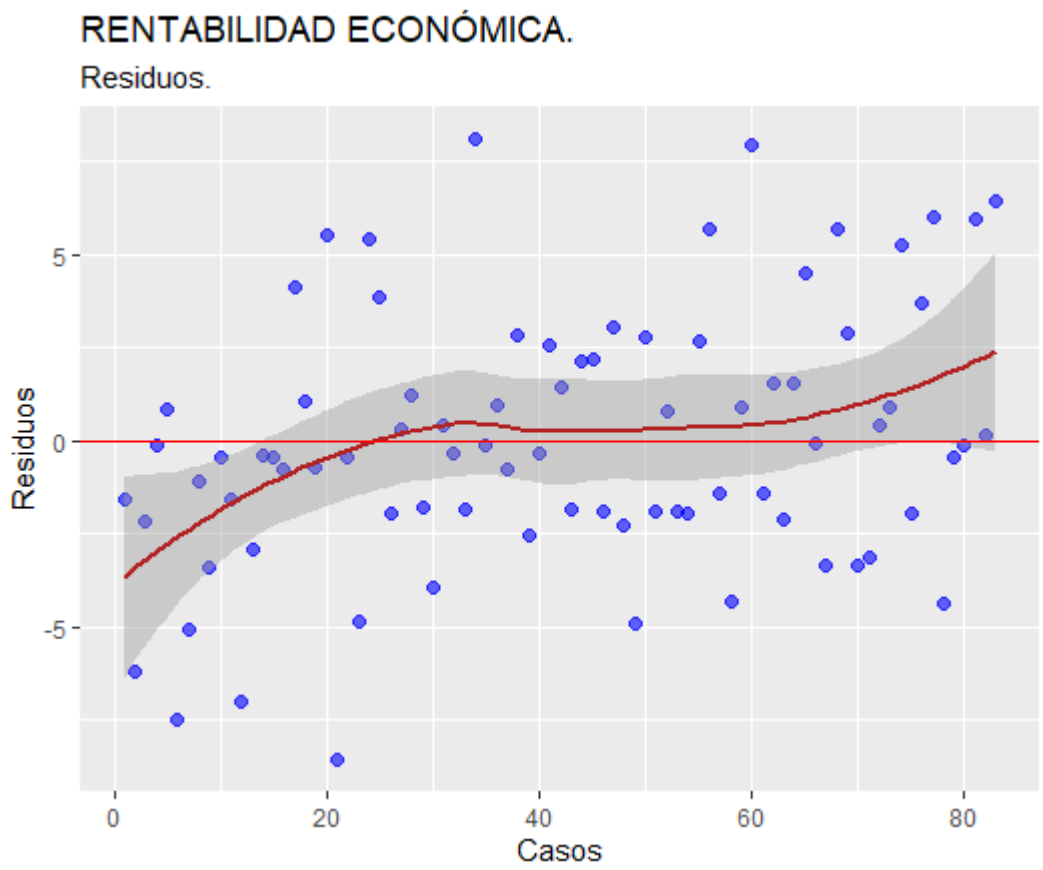
El resultado es:



En cuanto al gráfico de los residuos, el código es:

```
ggplot(data = resultados, aes(x = ORDEN, y = residuos)) +
  geom_point(size=2, alpha= 0.6, color = "blue") +
  geom_smooth(color = "firebrick") +
  geom_hline(yintercept = 0, color = "red")+
  ggtitle("RENTABILIDAD ECONÓMICA.", subtitle= "Residuos.")+
  xlab("Casos") +
  ylab("Residuos")
```

Y el gráfico:



En cuanto a la **forma funcional y linealidad**, se realiza la prueba de Ramsey-RESET. Para aplicar esta prueba hay que activar la librería lmtest.

```
check_model(ecuaDEF)

## Forma funcional y linealidad.

library (lmtest)
resettest(ecuaDEF, data= originales_so)
```

El resultado rechaza la hipótesis nula de forma funcional correcta (p-valor menor que 0,05). Posiblemente, y dada la baja bondad del modelo, este problema se deba a la **omisión de variables relevantes**.

```
        RESET test

data:  ecuaDEF
RESET = 11.031, df1 = 2, df2 = 77, p-value = 6.13e-05
```

La hipótesis de **rango pleno** (verificación de un posible problema de **multicolinealidad** entre las variables explicativas) ya ha sido tratado a lo largo de la práctica, mediante el análisis del *vif*.

Vamos a pasar a los supuestos referidos a la perturbación aleatoria. Concretamente, nos centraremos en la **normalidad** de la distribución de las perturbaciones aleatorias, y al contraste de **homoscedasticidad**, dado que el problema de autocorrelación es propio de los modelos estimados con datos de series temporales.

Para verificar el supuesto de **normalidad en las perturbaciones**, hemos de trabajar con los residuos del modelo, considerados como estimaciones de dichas perturbaciones. Estos residuos fueron guardados como la variable "residuos" del *data frame* "resultados".

Existen múltiples pruebas gráficas y contrastes para determinar si la distribución de los residuos sigue una ley Normal. La prueba gráfica más habitual es la del *gráfico qq* (cuantil-cuantil). Esta prueba compara los cuantiles teóricos que tiene una distribución normal con los que tienen la distribución de nuestra variable. No obstante, en esta práctica vamos a recurrir al **test de normalidad** de *Shapiro-Wilk*. Su código es:

```
## Normalidad de las perturbaciones.

shapiro.test(resultados$residuos)
```

El resultado obtenido es:

```
      Shapiro-Wilk normality test

data:  resultados$residuos
W = 0.98436, p-value = 0.4121
```

Por tanto, al ser el p-valor mayor que 0.05, **no** se rechaza la hipótesis nula de normalidad en las perturbaciones aleatorias.

Por último, vamos a contrastar la hipótesis básica de **homoscedasticidad en las perturbaciones aleatorias**. Existen varios contrastes estadísticos alternativos. Uno de los más utilizados es el de *Breush-Pagan*, habilitado en la librería lmtest, que ya fue activada. El código es:

```
## Homoscedasticidad

bptest(ecuaDEF)
```

El resultado obtenido es:

```
        studentized Breusch-Pagan test

data:  ecuaDEF
BP = 18.538, df = 3, p-value = 0.0003407
```

El p-valor lleva a rechazar la hipótesis nula de homoscedasticidad, por lo que el modelo puede estar afectado por un problema de heteroscedasticidad (varianza del vector de perturbaciones no constante), lo que podría llevar a una pérdida de eficiencia (precisión) de los estimadores de mínimos cuadrados ordinarios. El origen de este problema podría estar, de nuevo, en la omisión de una o varias variables relevantes en la especificación del modelo.

¿Y qué pasa con el cambio estructural?

Es difícil verificar la hipótesis de estabilidad o permanencia estructural, ya que el orden en el que aparecen los elementos de la muestra (empresas) es puramente aleatorio, o no tiene un criterio de ordenación único, como ocurre con las series temporales. No obstante, es cierto que en una misma muestra pueden existir diferentes patrones de relación entre variables dependientes y explicativas, derivados de la propia naturaleza de las variables, como de problemas técnicos (forma funcional incorrecta, omisión de variables relevantes, heteroscedasticidad, etc.) El cambio estructural puede hacer que las relaciones entre variables no queden bien plasmadas debido a unos mismos coeficientes estructurales estimados fijos no tienen capacidad para representar diferentes estructuras económicas reales, lo que se traduce en un problema de sesgadez, perdida de eficiencia e incluso inconsistencia en los estimadores MCO.

Un análisis previo que puede ser de gran utilidad para intentar identificar la existencia de varias estructuras de relaciones entre los elementos de la muestra puede ser el realizar un **análisis clúster** basado en las variables del modelo, lo cual puede permitir discriminar grupos de empresas (submuestras) con comportamientos diferentes entre sí, y homogéneas dentro de cada uno. Así, si se caracterizan estos grupos, se podría plantear, siempre que estos grupos *intra-sectoriales* tengan un tamaño suficiente, plantear una especificación y/o **estimación distinta para cada**

uno, o plantear la inclusión de variables dicotómicas o dummies para cada grupo de empresas, dentro de una única estimación.

Simulación.

Por último, vamos a utilizar el modelo estimado a efectos de simulación / previsión. Los datos del escenario están almacenados en el archivo de Microsoft® Excel® "eolica_100_mv_e.xlsx", hoja "Simula"; luego los importaremos mediante el código:

```
## PREVISIÓN

simula <- read_excel("eolica_100_mv_e.xlsx", sheet = "Simula")
simula <- data.frame(simula, row.names = 1)
summary (simula)
```

El *data frame* "Simula", con una sola fila, almacena el valor del escenario para las variables explicativas:

NOMBRE	LIQUIDEZ	MARGEN	SOLVENCIA	APALANCA	DIMENSION
Escenario A	1,50	60,76	50,74	75,02	GRANDE

El resultado de la previsión / simulación se guardará en el *data frame* "prevision", que se genera mediante la función predict(). Entre los argumentos, destaca el nombre del modelo con el que realizaremos la simulación ("object ="), el *data frame* donde se almacenan los valores de las variables explicativas ("newdata="), la opción de realizar una previsión por intervalo y, en tal caso, el intervalo de confianza. Posteriormente, se muestra en pantalla "prevision", y se crea el *data frame* "Resultado" como unión del *data frame* "simula" y del valor simulado, almacenado en el *data frame* "prevision":

```
prevision <-predict (object= ecuaDEF, newdata = simula,
interval="prediction", level=0.95)
prevision
Resultado <- cbind(simula, prevision)
```

Para terminar, utilizando las facilidades de los paquetes kable y kableExtra, pueden mostrarse en una tabla personalizada los resultados de la simulación:

```
Resultado %>%
  kable(caption = "Simulación / Previsión Modelo Rentabilidad
Económica", digits = c(0, 2, 2, 2, 2, 0, 2, 2, 2),
        col.names = c("Escenario", "Liquidez", "Margen", "Solvencia",
"Apalancamiento", "Dimensión", "Previsión", "Inferior 95%", "Superior
95%")) %>%
  kable_styling(full_width = F, bootstrap_options = "striped",
"bordered", "condensed", position = "center", font_size = 12) %>%
  row_spec(0, bold= T, align = "c") %>%
  row_spec(1:(nrow(Resultado)), bold= F, align = "c")
```

El resultado es:

Simulación / Previsión Modelo Rentabilidad Económica

Escenario	Liquidez	Margen	Solvencia	Apalancamiento	Dimensión	Previsión	Inferior 95%	Superior 95%
Escenario A	1.5	60.76	50.74	75.02	GRANDE	6.57	-0.56	13.69

Según la simulación, para el escenario previsto la Rentabilidad Económica esperada es del 6,57%, o estará situada entre el -0,56% y el 13,69% con un nivel de confianza del 95%.

Updated: 29/11/2022