# Regresión: Modelos Estadísticos

Conjunto de Datos: Cheddar Faraway

true true true true

25/03/2022

**Abstract**

Hemos analizado con las herramientas proporcionadas en el curso de Modelos Estadísticos el conjunto de datos, *Cheddar*, distribuido en la librería Faraway de R. Para ello hemos utilizado diversas técnicas de regresión lineal y no lineal.

## 1 Introducción

En un estudio de queso Cheddar realizado en el Valle de Latrobe (Victoria, Australia), se estudiaron muestras de queso en las que se analizó su composición química y fueron dadas a probar a distintos sujetos para que valoraran su sabor. Los valores asignados a cada queso son el resultado de combinar las distintas valoraciones.

El DataFrame **cheddar** de la librería **faraway** consiste de 30 muestras de queso Cheddar en las que se ha medido el sabor (*taste*) y las concentraciones de ácido acético (*Acetic*), ácido sulfhídrico (*H2S*) y lactosa (*Lactic*).

Tenenemos un conjunto de datos en el que se recogen observaciones de una cata de quesos, nuestras variables son:

- **Taste:** una valoración subjetiva de los jueces.
- **Acetic:** la concentración de ácido acético en un queso de terminado en esca la logarítmica
- **H2S:** la concentración de sulfito de hidrógeno en escala logarítmica.
- **Lactic:** Concentración de ácido láctico

A lo largo del documento hacemos uso de las siguientes librerias de R:

```
library(faraway)
library(leaps)
library(MASS)
library(PASWR)
library(car)
library(ggplot2)
library(plyr)
library(GGally)
library(corrplot)
library(plotly)
library(scatterplot3d)
library(cowplot)
```

Vamos a utilizar el dataset *Cheddar*. Cargamos los datos y enseñamos las primeras observaciones.

```r
# load cheddar cheddar
data(cheddar)
head(cheddar)
```

```
##   taste Acetic   H2S Lactic
## 1  12.3  4.543 3.135   0.86
## 2  20.9  5.159 5.043   1.53
## 3  39.0  5.366 5.438   1.57
## 4  47.9  5.759 7.496   1.81
## 5   5.6  4.663 3.807   0.99
## 6  25.9  5.697 7.601   1.09
```

Si en nuestro dataset tuviesemos entradas vacías (NA), tenemos varias posibilidades para lidiar con este problema:

- No utilizar/Eliminar las observación que contienen valores.
- No utilizar/Eliminar las variables que contienen las entradas vacías.
- Intentar completar los valores. Existen métodos menos y más sofisitcados:
  - Remplezar con la **media, media o moda**.
  - Crear una **nueva categoría** para valores vacíos.
  - Utilizar algún modelo de **regresión**.
  - Usar un modelo de **K-Nearest Neighbors (KNN)**.

A continuación, comprobamos que no hay entradas vacías, s

```r
# check if any entry has a missing value (NA)
any(is.na(cheddar))
```

```
## [1] FALSE
```

```r
sapply(cheddar, class)
```

```
##     taste    Acetic       H2S    Lactic
## "numeric" "numeric" "numeric" "numeric"
```

Toas las variables son numéricas (**cuantitativas**). No hay que transformar las variables no cuantitativas (**cualitativas**), conviertiendolas en variables binarias. Para ello, podríamos deberíamos hacer encoding a variables binarias, el lenguage de programación R nos permite utilizar *as.numeric*.

Con estas variables vamos a intentar **explicar** cómo los valores observados de una variable Y (taste) dependen de los valores de otras variables (Acetic, H2S, Latcic), a través de una relación funcional del tipo Y = f(X) También vamos a intentar **predecir** el valor de la variable Y para valores no observados de las variables X.

Usamos el número de observaciones para determinar los conjuntos de train y test.

```r
numObs <- dim(cheddar)[1]
numObs
```

```
## [1] 30
```

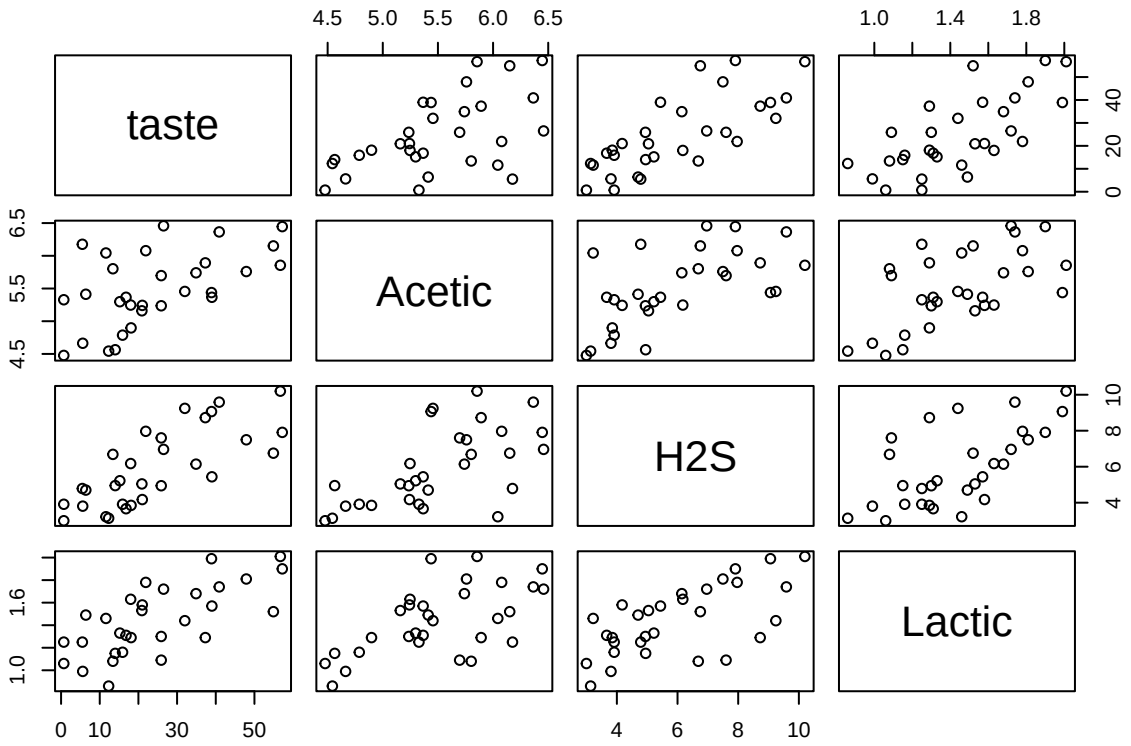
Tenemos 30 observaciones en nuestro dataset. Ahora procedemos a divirlo en el *conjunto de train y test*. El primero lo utilizaremos para entrenar nuestros modelos y el segundo lo usamos para cuantificar el error de los modelos.

Ahora nos hacemos las siguientes preguntas: ¿Podemos suponer que la distribución de las variables es normal?,¿Tenemos alguna en la que falten datos?, ¿Tenemos *outliers*?, etc. En las siguientes secciones trataremos de responder a estas preguntas y muchas otras acercar de nuestro conjunto de datos.

```
# Para asegurar que sea reproducible utilizamos una semilla, que permite fijar los valores pseudoaleato
set.seed(1)
train <- sample(c(TRUE, FALSE),
                size = nrow(cheddar),
                replace = TRUE,
                prob = c(0.7, 0.3))
test <- !train
```

Hacemos un pequeño estudio preliminar de nuestras variables. Mostramos un scatter plot de las cada variable contrastada con el resto. Esto permite ver *a ojo* si algún par de variables tiene correlación.

```
plot(cheddar)
```




Ahora, utilizamos la funcón *summary* de R, la cual nos permite estimar algunos de las características de la distribución del dataset. La siguiente tabla nos muestra los estadísticos más comunes: el mínimo, máximo, mediana, media y el 1er y 3er cuartil.

```
summary(cheddar) # Distribución de las variables
```

```
##      taste           Acetic           H2S             Lactic      
##  Min.   : 0.70   Min.   :4.477   Min.   : 2.996   Min.   :0.860  
##  1st Qu.:13.55   1st Qu.:5.237   1st Qu.: 3.978   1st Qu.:1.250  
##  Median :20.95   Median :5.425   Median : 5.329   Median :1.450  
##  Mean   :24.53   Mean   :5.498   Mean   : 5.942   Mean   :1.442  
##  3rd Qu.:36.70   3rd Qu.:5.883   3rd Qu.: 7.575   3rd Qu.:1.667  
##  Max.   :57.20   Max.   :6.458   Max.   :10.199   Max.   :2.010
```

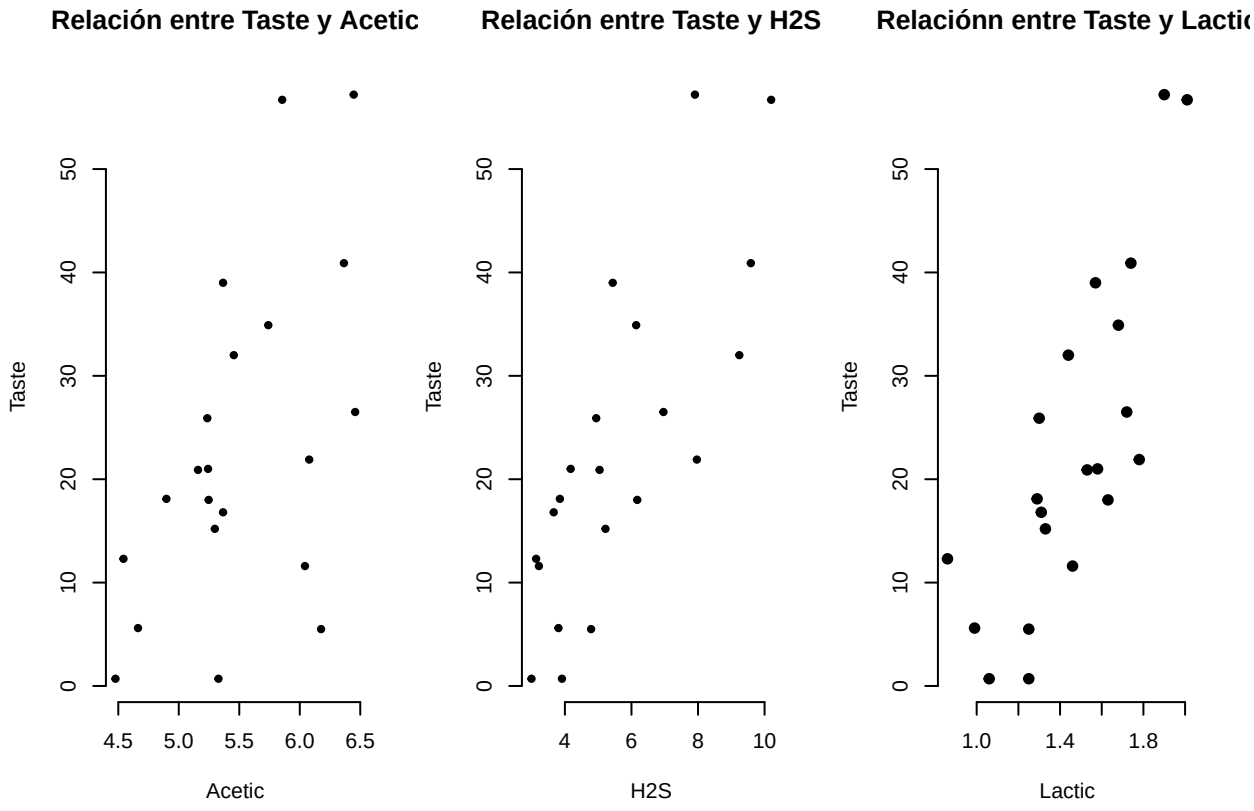
Ploteamos las gráficas de dispersion entre la variable respuesta *taste* y las variables predictoras *Acetic, H2S, Lactic.*

```r
layout(matrix(1:3, nrow = 1))

# plot taste ~ Acetic
plot(Acetic, taste,
     main = "Relación entre Taste y Acetic",
     xlab = "Acetic", ylab = "Taste",
     pch = 20, frame = FALSE)

# plot taste ~ H2S
plot(H2S, taste,
     main = "Relación entre Taste y H2S",
     xlab = "H2S", ylab = "Taste",
     pch = 20, frame = FALSE)

# plot taste ~ Lactic
plot(Lactic, taste,
     main = "Relaciónn entre Taste y Lactic",
     xlab = "Lactic", ylab = "Taste",
     pch = 19, frame = FALSE)
```




Podemos observar que la que aperentemente guarda una menor relación lineal con taste es la variable Acetic, esto será comprobado con distintos tests.

# 2 Estudio y evaluación del modelo completo.

Simplificamos nuestra notación para las variables. Intentaremos predecir la variable *taste* usando el resto de variables. Para empezar, definimos el modelo completo, el cual se usan todas las variables para nuestro modelo lineal múltiple.

$$Y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_{p-1} x_{i(p-1)} + \epsilon_i, \quad i = 1, \ldots, n$$

donde $Y_i$ es el valor de la variable respuesta para el individuo $i$-ésimo,
$\beta_0$ y los $\beta_j$ son los par´ametros $j = 1, ..., p-1$,
$x_{ij}$ son los elementos de la matriz de las variables explicativas
$\epsilon_i$ es el término del error aleatorio que suponemos que se distribuye como una $\mathcal{N}(0, \sigma^2)$, donde $\sigma^2$ es la varianza que suele ser desconocida.

## 2.1 Resolución mediante matrices

Utilizamos el método de mínimos cuadrados que estima los valores $\hat{\beta}$ intentando minimizar los errores $\epsilon$. Como hemos visto en clase, la fórmula que se deduce es:

$$\hat{\beta} = (X^t X)^{-1} X^t Y$$

donde $X$ es una columna de 1's concatenada con las variables que usamos para predecir. Es importante clarificar que en este proceso solo usamos el training set.

```
x <- model.matrix( ~ Acetic + H2S + Lactic, data = cheddar[train,])
betahat <- solve(crossprod(x, x), crossprod(x, taste))
betahat <- c(betahat)
betahat
```

```
## [1] -14.016212  -5.066965   3.217036  31.938449
```

Por tanto, aproximamos las $\beta$ modelo lineal completo con los valores de $\hat{\beta}_0, \ldots, \hat{\beta}_3$ con los siguientes valores:

$$\hat{\beta}_0 = -14.016212, \quad \hat{\beta}_1 = -5.066965, \quad \hat{\beta}_2 = 3.217036, \quad \hat{\beta}_3 = 31.938449$$

## 2.2 Resolución usando librerias de R

Podemos utilizar la función *lm*, ya programada en R.

```
model.all.lm <- lm(taste ~ ., data = cheddar[train,])
model.all.lm$coefficients # Coeficientes del modelo completo
```

```
## (Intercept)      Acetic          H2S      Lactic
##  -14.016212   -5.066965    3.217036   31.938449
```

Evidentemente los resultados son los mismos.

Estudiemos preliminarmente si es un modelo lineal adecuado, para ello comprobaremos las hipótesis estándar del modelo lineal de regresión usando el **test de normalidad Shapiro-Wilk**. La función *shapiro.test* le pasamos por parámetro el residuo/error de cada una de las muestras y nos devuelve un $p$-valor.

```
shapiro.test(resid(model.all.lm))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(model.all.lm)
## W = 0.97755, p-value = 0.8865
```

Observamos que estamos en la hipótesis de que el error nuestro modelo se distribuye de manera normal, ya que el p-valor es $0.8865 > 0.05$.

Ahora nos preguntamos si hay variables que tienen un mayor impacto en el modelo. Estas variables podrían ser *outliers* y podríamos deshecharlas del training set ya que podrían estar perjudicando la predicción del modelo negativamente.

```
summary(model.all.lm)
```

```
##
## Call:
## lm(formula = taste ~ ., data = cheddar[train, ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.7641  -4.0925  -0.7499   4.3790  17.7545
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -14.016     20.490  -0.684   0.5032
## Acetic        -5.067      5.120  -0.990   0.3363
## H2S            3.217      1.472   2.185   0.0432 *
## Lactic        31.938     12.852   2.485   0.0237 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.109 on 17 degrees of freedom
## Multiple R-squared:  0.7251, Adjusted R-squared:  0.6765
## F-statistic: 14.94 on 3 and 17 DF,  p-value: 5.103e-05
```

Obervamos el p-valor de A nos indica que con casi toda seguridad A no tiene impacto real en el modelo ( $> 0.94$)

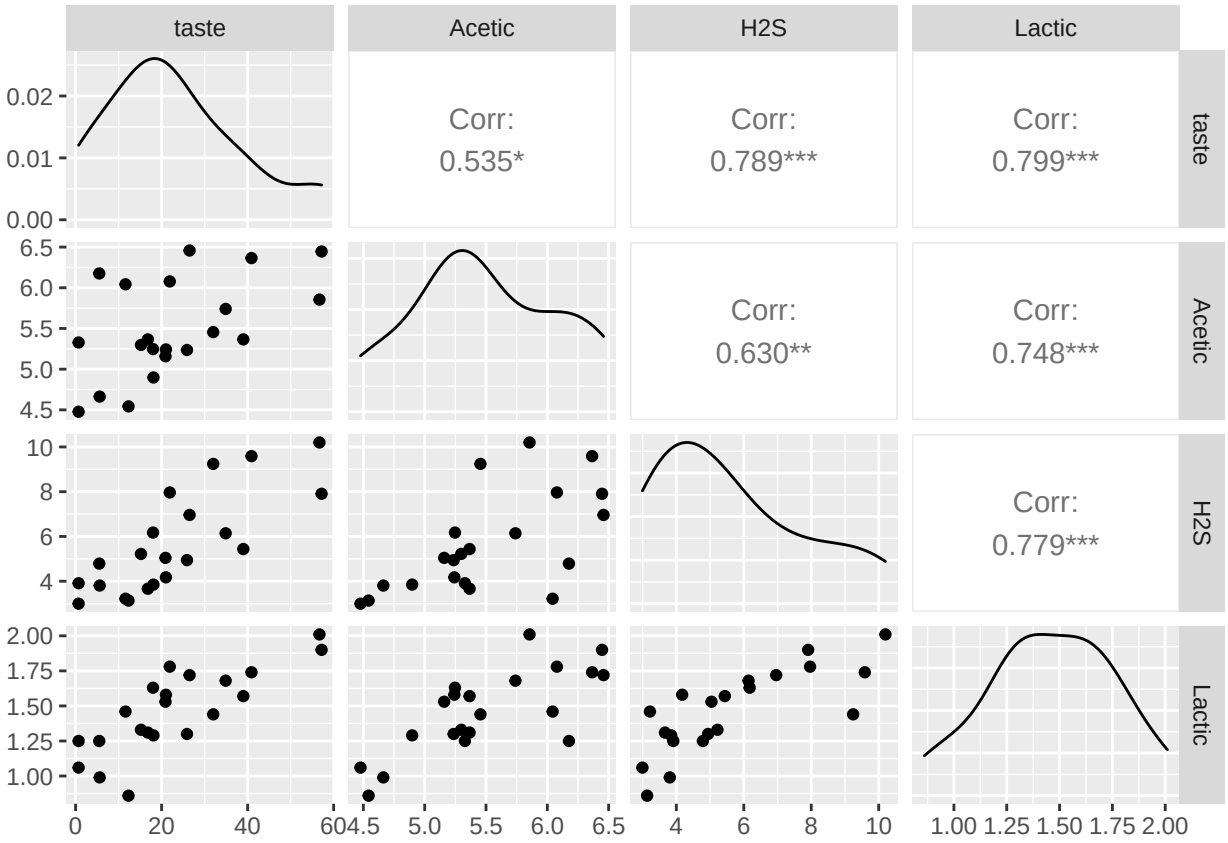
Los p-valores son lo suficientemente bajos como para rechazar varianza constante

Una vez hemos concluido que aunque estamos en las hipótesis de regresión lineal el modelo completo a pesar de ser el más complejo probablemente da resultados similares a otro más simple.

#### 2.2.0.1 Correlaciones y tabla de resultados con el estudio de sus p-valores

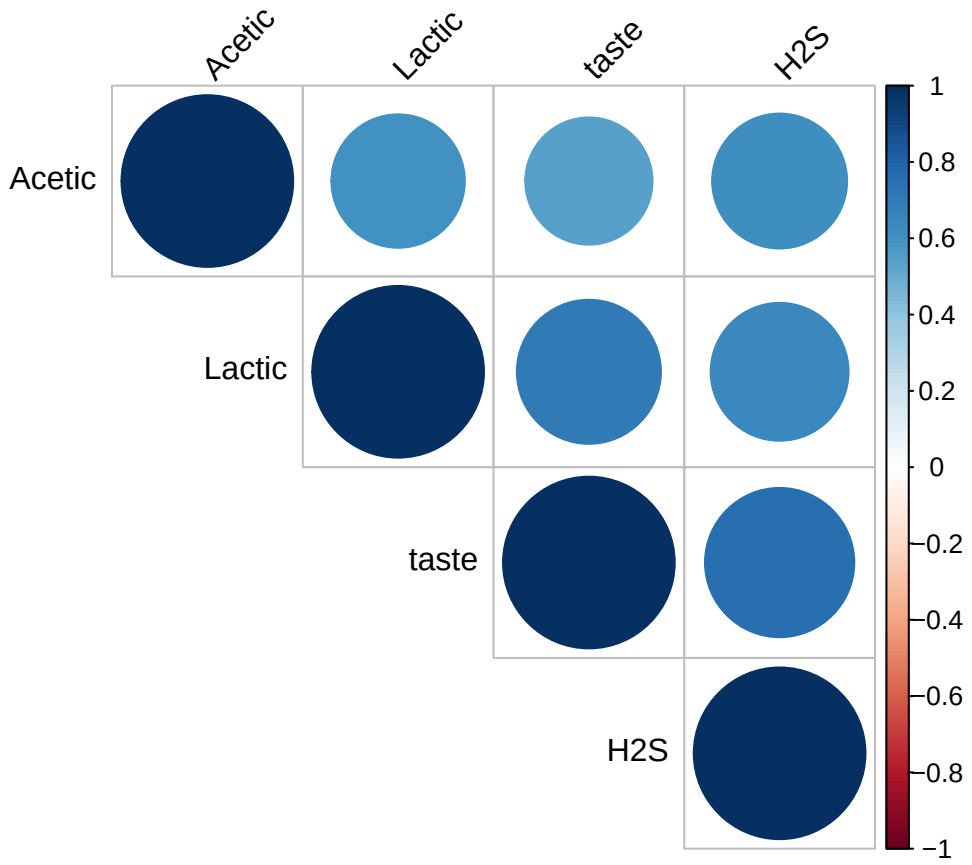
Usamos el paquete *GGplot* de R, el cual nos permite visualizar la correlación y dispersión entre las distintas variables.

```
corplot <- ggpairs(cheddar[train,], progress=FALSE)
corplot
```




Para medir la correlación podemos utilizar también la función *cor*, que utiliza el método de **Pearon**. Podemos observar que la correlación de la propia variable con sigo misma es máxima (=1).

```
mat_cor <- cor(cheddar, method = "pearson")
corrplot(mat_cor, type = "upper", order = "hclust", tl.col = "black", tl.srt = 45)
```




Ahora utilizamos el anális **anova** (Analysis of Variance), para justificar si podemos eliminar alguna variable del modelo.

```
anova(model.all.lm)
```

```
## Analysis of Variance Table
##
## Response: taste
##           Df Sum Sq Mean Sq F value    Pr(>F)
## Acetic     1 1466.8 1466.80 17.6761 0.0005961 ***
## H2S        1 1740.8 1740.83 20.9783 0.0002660 ***
## Lactic     1  512.5  512.50  6.1761 0.0236545 *
## Residuals 17 1410.7   82.98
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Todos nuestros p-valores son adecuados a un nivel $\alpha = 0.05$. En nuestro caso nos vale, sin embargo, la variable *Lactic* no lo cumpliría si disminuimos el nivel de $\alpha$ a una cota inferior como 0.01.

## 2.3 ¿Tiene *outliers* nuestra muestra?

Para comprobarlo basta realizar el **test de Bonferroni** sobre nuestro modelo completo:

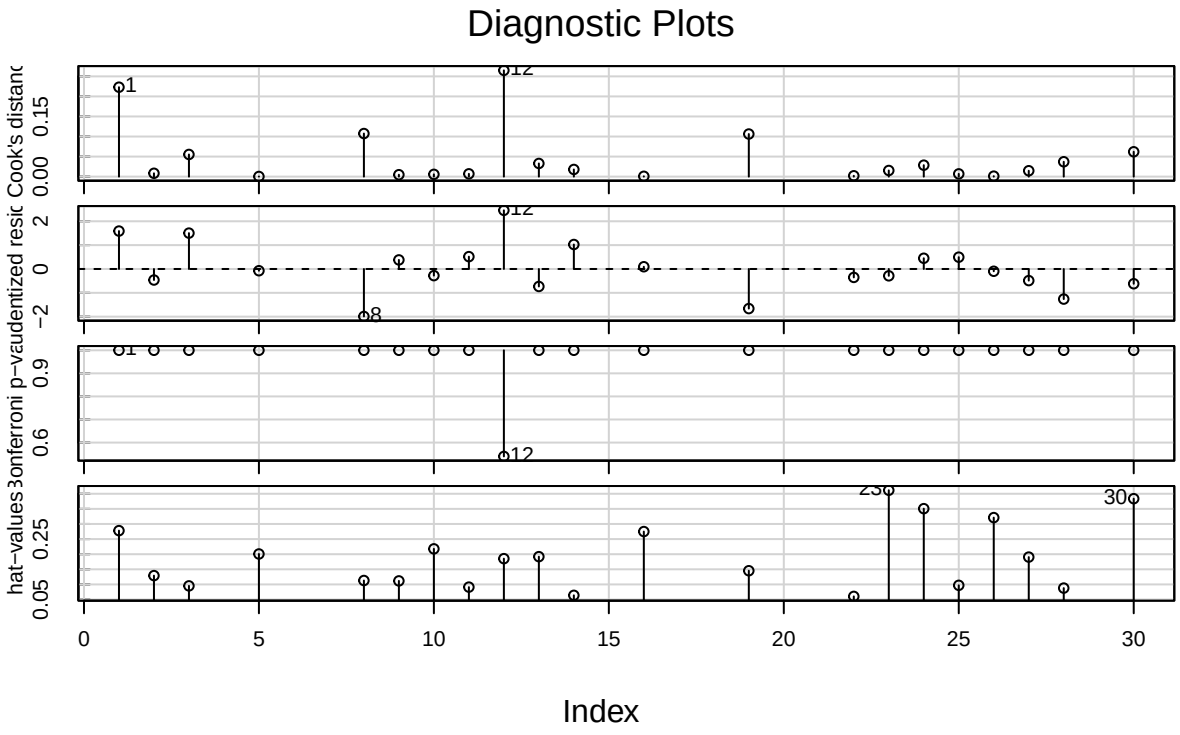
```
outlierTest(model.all.lm)
```

```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##    rstudent unadjusted p-value Bonferroni p
## 12 2.458893           0.025711      0.53993
```

Concluimos con un nivel $\alpha = 0.05$ que no tenemos ningún outlier en nuestra. Lo más cercano a un *outlier* que tenemos es la observación número 12, que tiene un valor **Bonferroni p** de 0.53993 (no se acerca a 0.05). Por tanto, no tenemos razones por las que eliminar alguna observación inusual de nuestro conjunto de datos.

Esto se puede comprobar graficamente a través del siguiente gráfico, el cual mide la influencia de cada observación sobre cada una de las betas de nuestro modelo. Vemos que la que más influye es la antes mencionada obsercación 12.

```
influenceIndexPlot(model.all.lm)
```




Podemos ver que hay un pequeño grupo de observaciones que aunque no sean *outliers* pueden alterar la claridad del modelo.

## 2.4 ¿Cuál es el mejor modelo?

Como dice el **Principio de la Navaja de Ockham**, a menudo la explicación más simple es la correcta. Queremos seleccionar predictores que explican los datos de la manera más simple posible, sin disminuir la calidad de las predicciones mucho.

### 2.4.1 Método Backward

Partimos del modelo completo estudiado en la sección anterior y aplicamos con $\alpha = 0.05$, el metodo de Backward, que consiste en eliminar la variable que *menos influya* a la predicción.

```
drop1(model.all.lm, test = "F")
```

```
## Single term deletions
##
## Model:
```

```
## taste ~ Acetic + H2S + Lactic
##        Df Sum of Sq    RSS     AIC F value  Pr(>F)
## <none>              1410.7  96.354
## Acetic  1     81.26 1492.0  95.530  0.9792 0.33627
## H2S     1    396.17 1806.9  99.551  4.7741 0.04318 *
## Lactic  1    512.50 1923.2 100.862  6.1761 0.02365 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Quitamos Acetic del modelo debido que su p-valor es $> 0.05$

```
model.backward <- update(model.all.lm, . ~ . - Acetic)
drop1(model.backward, test = "F")
```

```
## Single term deletions
##
## Model:
## taste ~ H2S + Lactic
##        Df Sum of Sq    RSS    AIC F value  Pr(>F)
## <none>              1492.0 95.530
## H2S     1    361.58 1853.5 98.087  4.3624 0.05122 .
## Lactic  1    443.44 1935.4 98.994  5.3499 0.03275 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Repetimos el proceso con la variable H2S, ya que tiene un valor mayor que $\alpha = 0.05$

```
model.backward <- update(model.backward, . ~ . - H2S)
drop1(model.backward, test = "F")
```

```
## Single term deletions
##
## Model:
## taste ~ Lactic
##        Df Sum of Sq    RSS     AIC F value    Pr(>F)
## <none>              1853.5  98.087
## Lactic  1    3277.3 5130.8 117.469  33.594 1.388e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

El p-valor es menor que $\alpha = 0.05$. Por tanto, hemos concluido, ya que no tenemos suficiente certeza para poder eliminar otra variable.

Por tanto, tenemos como resultado que la variable que mejor explica el *taste* es *Lactic*.

```
# Modelo resultante del método Backward
model.backward
```

```
##
## Call:
## lm(formula = taste ~ Lactic, data = cheddar[train, ])
##
## Coefficients:
## (Intercept)       Lactic
##      -39.82        42.95
```

Modelo resultante: **taste ~ Lactic.**

### 2.4.2 Método Forward

El método Forward consiste en empezar con un modelo de una variable y vamos añadiendo las que más influyan. De esta manera tenemos:

```
SCOPE<-(~.+Acetic + H2S + Lactic)
model.forward.lm <- lm(taste~1,data= cheddar[train,])
add1(model.forward.lm,scope=SCOPE,test="F")
```

```
## Single term additions
##
## Model:
## taste ~ 1
##         Df Sum of Sq    RSS     AIC F value    Pr(>F)
## <none>               5130.8 117.469
## Acetic  1    1466.8 3664.0 112.398  7.6062   0.01252 *
## H2S     1    3195.4 1935.4  98.994 31.3700 2.115e-05 ***
## Lactic  1    3277.3 1853.5  98.087 33.5944 1.388e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Actualizamos añadiendo Lactic por tener el menor p-valor.

```
model.forward.lm <- update(model.forward.lm, .~. + Lactic)
add1(model.forward.lm,scope=SCOPE,test="F")
```

```
## Single term additions
##
## Model:
## taste ~ Lactic
##         Df Sum of Sq    RSS    AIC F value  Pr(>F)
## <none>               1853.5 98.087
## Acetic  1     46.68 1806.9 99.551  0.4650 0.50399
## H2S     1    361.58 1492.0 95.530  4.3624 0.05122 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Con nivel de significación $\alpha = 0.05$ este sería nuestro modelo final.

```
model.forward.lm
```

```
##
## Call:
## lm(formula = taste ~ Lactic, data = cheddar[train, ])
##
## Coefficients:
## (Intercept)       Lactic
##      -39.82        42.95
```

Modelo resultante: **taste** ~ **Lactic.**

### 2.4.3 Observación

A este modelo, **taste** ~ **Lactic** le vamos a dar un nombre, posteriormente estudiaremos los problemas que presenta.

```
model.final1 <- lm(taste ~ Lactic, data = cheddar[train,])
```

## 2.5 Construcción por criterios

En esta subsección trataremos de encontrar un candidato a mejor modelo, construyendo nuestros modelos usando distintos enfoques.

### 2.5.1 $R^2$ ajustado

Es un criterio de optimización que se utiliza en la construcción de modelos, se basa en el estadísitico $R^2$.

```
models <- regsubsets(taste ~ ., data = cheddar[train,])
summary(models)
```

```
## Subset selection object
## Call: regsubsets.formula(taste ~ ., data = cheddar[train, ])
## 3 Variables  (and intercept)
##         Forced in Forced out
## Acetic      FALSE      FALSE
## H2S         FALSE      FALSE
## Lactic      FALSE      FALSE
## 1 subsets of each size up to 3
## Selection Algorithm: exhaustive
##          Acetic H2S Lactic
## 1  ( 1 ) " "    " " "*"
## 2  ( 1 ) " "    "*" "*"
## 3  ( 1 ) "*"    "*" "*"
```

```
MR2adj <- summary(models)$adjr2
MR2adj
```

```
## [1] 0.6197312 0.6769081 0.6765346
```

```
which.max(MR2adj)
```

```
## [1] 2
```

```
summary(models)$which[which.max(MR2adj), ]
```

```
## (Intercept)      Acetic         H2S      Lactic
##        TRUE       FALSE        TRUE        TRUE
```

Modelo resultante: **taste ~ H2S + Lactic.**

### 2.5.2 Criterio de $C_p$ de Mallow

Este criterio se basa en el estadístico de Mallow $C_p$.

```
MCp <- summary(models)$cp
MCp
```

```
## [1] 5.336570 2.979216 4.000000
```

```
which.min(MCp)
```

```
## [1] 2
```

```
summary(models)$which[which.min(MCp), ]
```

```
## (Intercept)      Acetic         H2S      Lactic
##        TRUE       FALSE        TRUE        TRUE
```

Modelo resultante: **taste ~ H2S + Lactic.**

### 2.5.3 Criterio de Informacion de Bayes (BIC)

```
MBIC <- summary(models)$bic
MBIC
```

```
## [1] -15.29253 -16.80519 -14.93673
```

```
which.min(MBIC)
```

```
## [1] 2
```

```
summary(models)$which[which.min(MBIC), ]
```

```
## (Intercept)      Acetic         H2S      Lactic
##        TRUE       FALSE        TRUE        TRUE
```

Modelo resultante: **taste ~ H2S + Lactic.**

### 2.5.4 Criterio de Informacion de Akaike (AIC)

```
stepAIC(model.all.lm, scope = SCOPE, k = 2)
```

```
## Start:  AIC=96.35
## taste ~ Acetic + H2S + Lactic
##
##          Df Sum of Sq    RSS     AIC
## - Acetic  1     81.26 1492.0  95.530
## <none>                1410.7  96.354
## - H2S     1    396.17 1806.9  99.551
## - Lactic  1    512.50 1923.2 100.862
##
## Step:  AIC=95.53
## taste ~ H2S + Lactic
##
##          Df Sum of Sq    RSS    AIC
## <none>                1492.0 95.530
## + Acetic  1     81.26 1410.7 96.354
## - H2S     1    361.58 1853.5 98.087
## - Lactic  1    443.44 1935.4 98.994
```

```
##
## Call:
## lm(formula = taste ~ H2S + Lactic, data = cheddar[train, ])
##
## Coefficients:
## (Intercept)          H2S       Lactic
##     -31.122        3.054       25.210
```

Modelo resultante: **taste ~ H2S + Lactic.**

Nótese que los modelos obtenidos por metodos de criterios coinciden.

```
model.crit <- lm(taste ~ H2S + Lactic, data=cheddar[train,])
anova(model.crit, model.all.lm)
```

```
## Analysis of Variance Table
##
## Model 1: taste ~ H2S + Lactic
## Model 2: taste ~ Acetic + H2S + Lactic
```

```
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     18 1492.0
## 2     17 1410.7  1    81.258 0.9792 0.3363
```

Con un valor de 0.3363 no podemos decir que los modelos sean signficativamente diferentes con ningún nivel $\alpha$ habitual, interpretamos que el segundo modelo aunque más simple, no es significativamente peor. ## Observación

A este , **taste** ~ **H2S + Lactic** le vamos a dar un nombre, en la siguiente sección lo estudiaremos con detalle.

```
model.final2 <- lm(taste ~ H2S + Lactic, data = cheddar[train,])
```

# 3 Diagnóstico

En esta sección estudiaremos si nuestros *modelos* cumplen las condiciones necesarias de un modelo de regresión lineal.

Nuestro enfoque consistirá en un analisis gráfico, acompañado de tests estadísticos en los casos en los que se aprecie una discrepancia notable.

## 3.1 ¿Son nuestros *modelos*, modelos de regresión lineal?: Comprobación de hipótesis.

En la sección 3 se toma un enfoque *naïve* a la hora de construir los modelos, ya que no hemos estudiado si hay observaciones influyentes, podríamos tener una muestra que no es la adecuada para el estudio de nuestros datos.

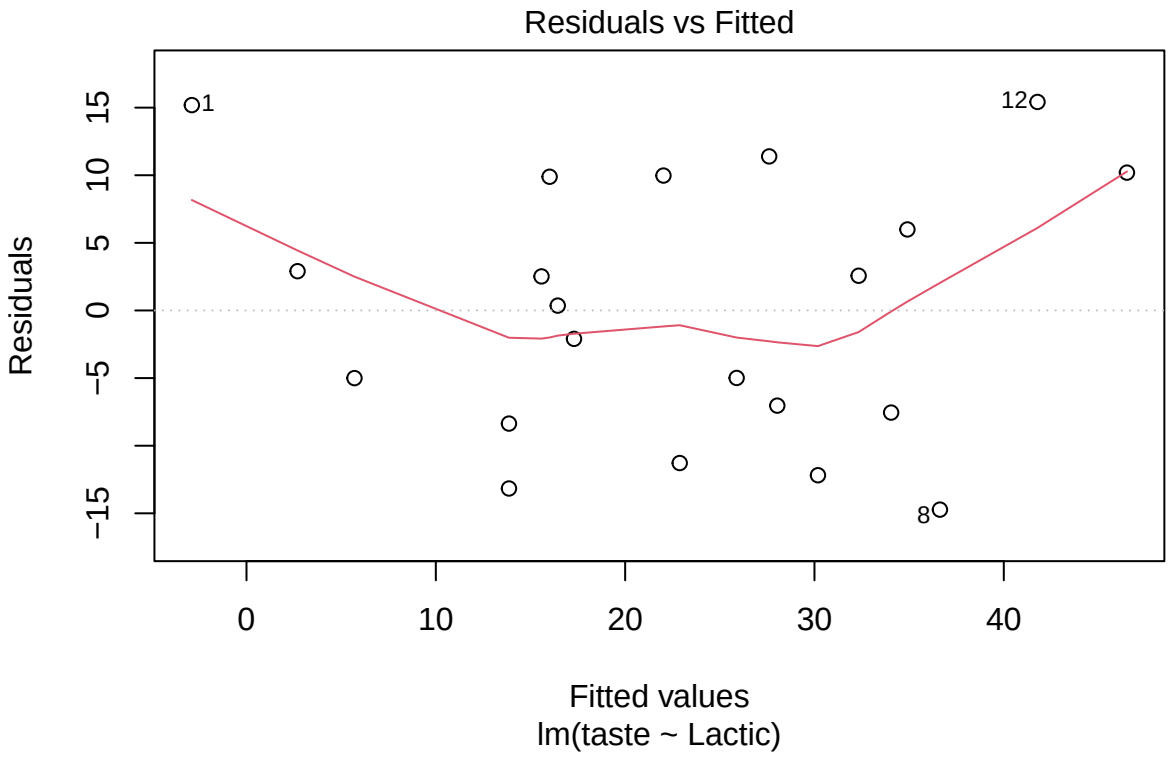
Un modelo de regresión lineal debe satisfacer las siguientes hipótesis con nivel de siginificación $\alpha$ adecuado:

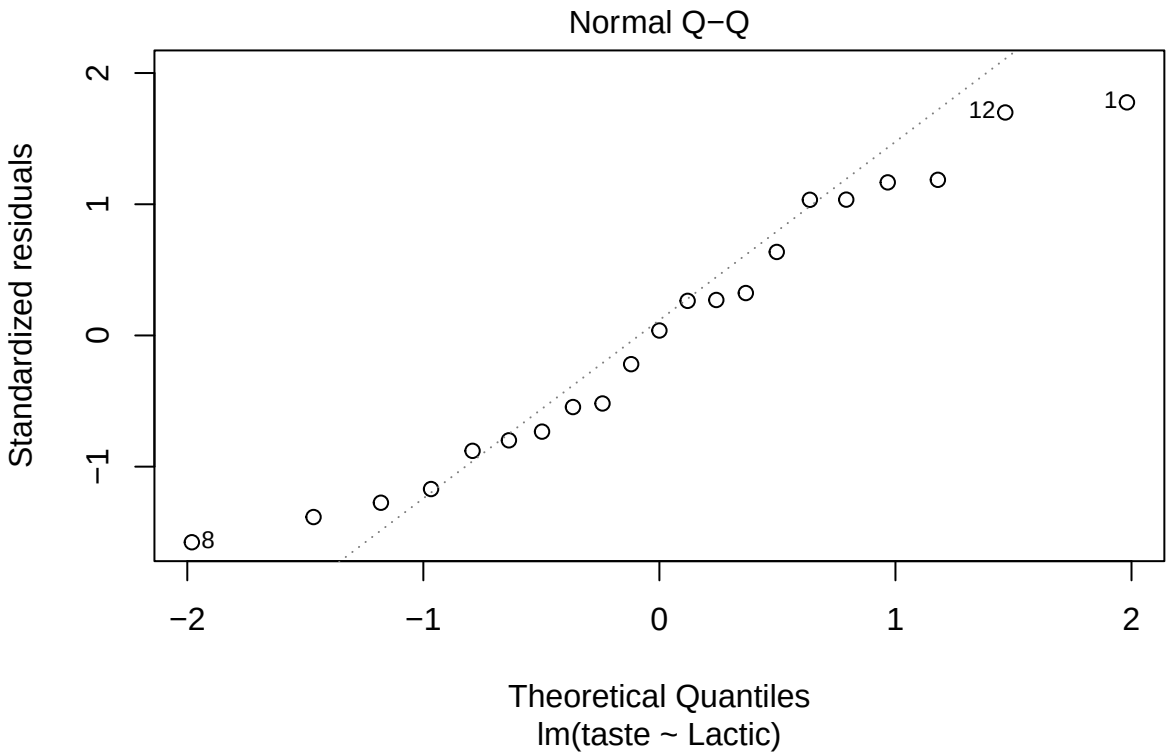
1. Los errores $\epsilon_i$ tienen distribución normal.
2. Los errores $\epsilon_i$ tienen media cero.
3. Los errores $\epsilon_i$ tienen varianza constante.
4. Los errores $\epsilon_i$ no están correlacionados.

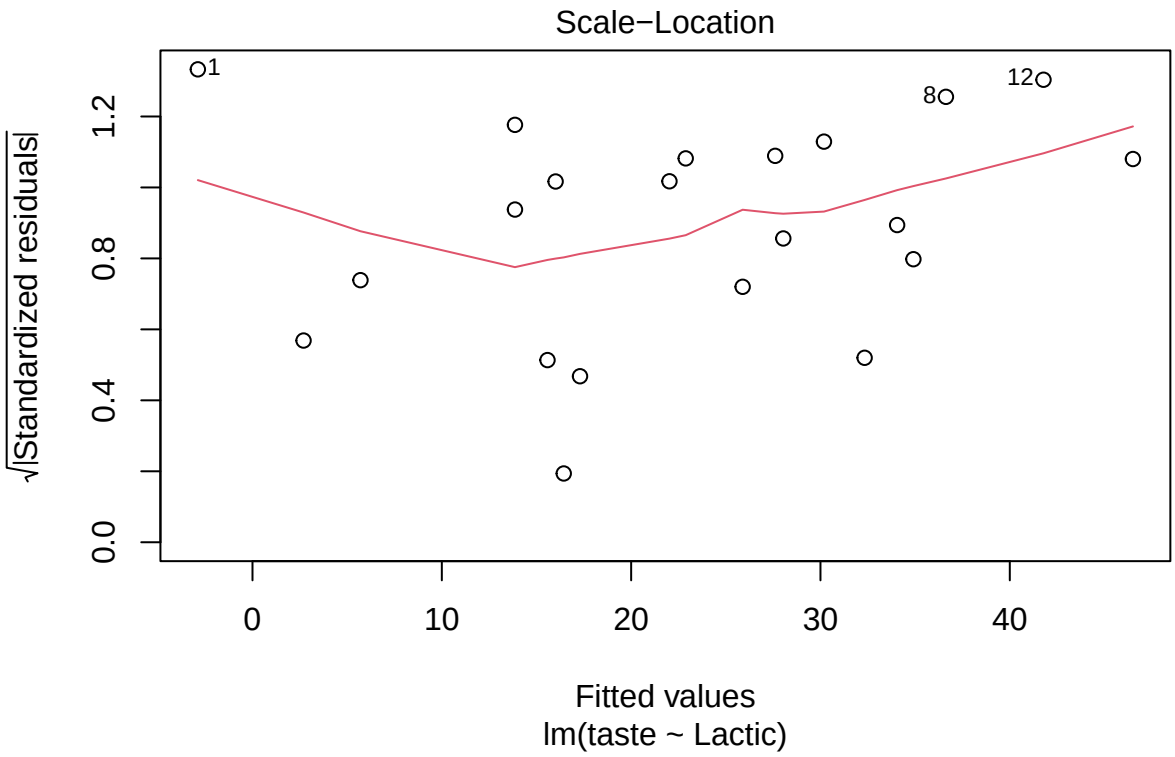
### 3.1.1 Estudios preliminares

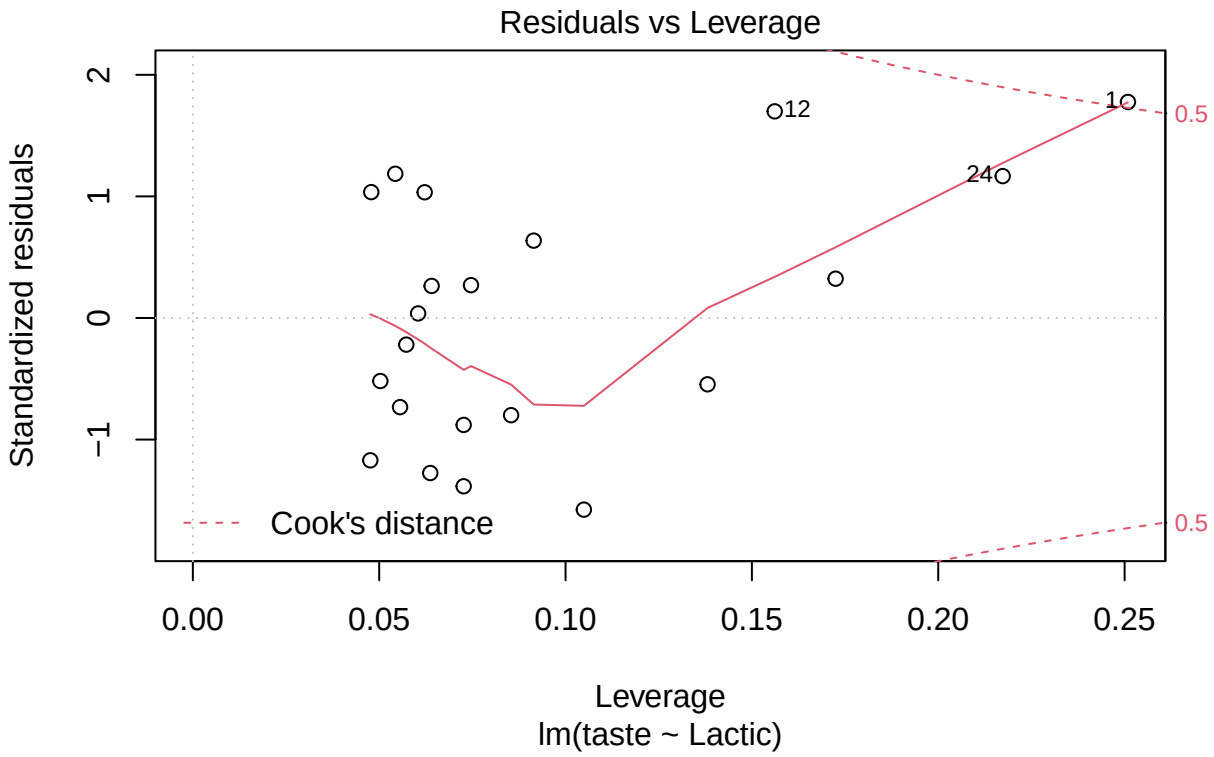
### 3.1.2 Model.final1

```
plot(model.final1)
```


Residuals vs Fitted
Residuals
Fitted values
lm(taste ~ Lactic)
1
12
8
15
10
5
0
−5
−15
0
10
20
30
40


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(taste ~ Lactic)
8
12
1
−2
−1
0
1
2


Scale−Location
√|Standardized residuals|
0.0
0.4
0.8
1.2
1
8
12
0
10
20
30
40
Fitted values
lm(taste ~ Lactic)

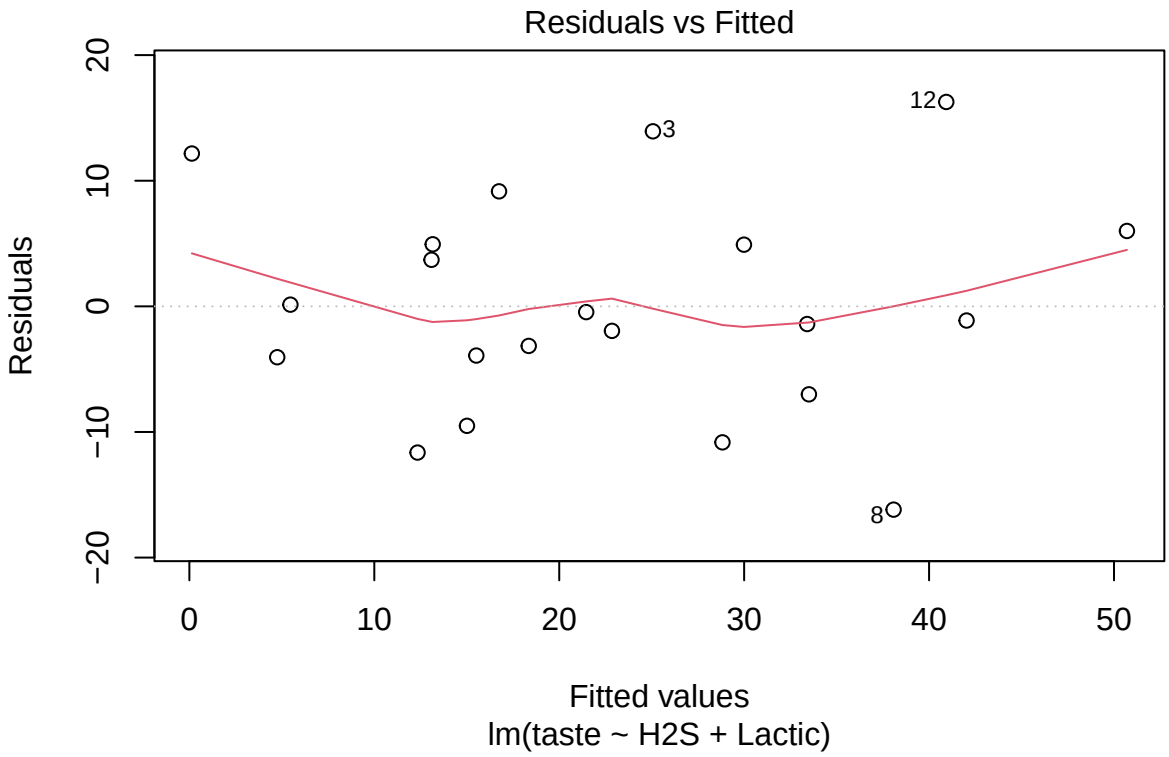


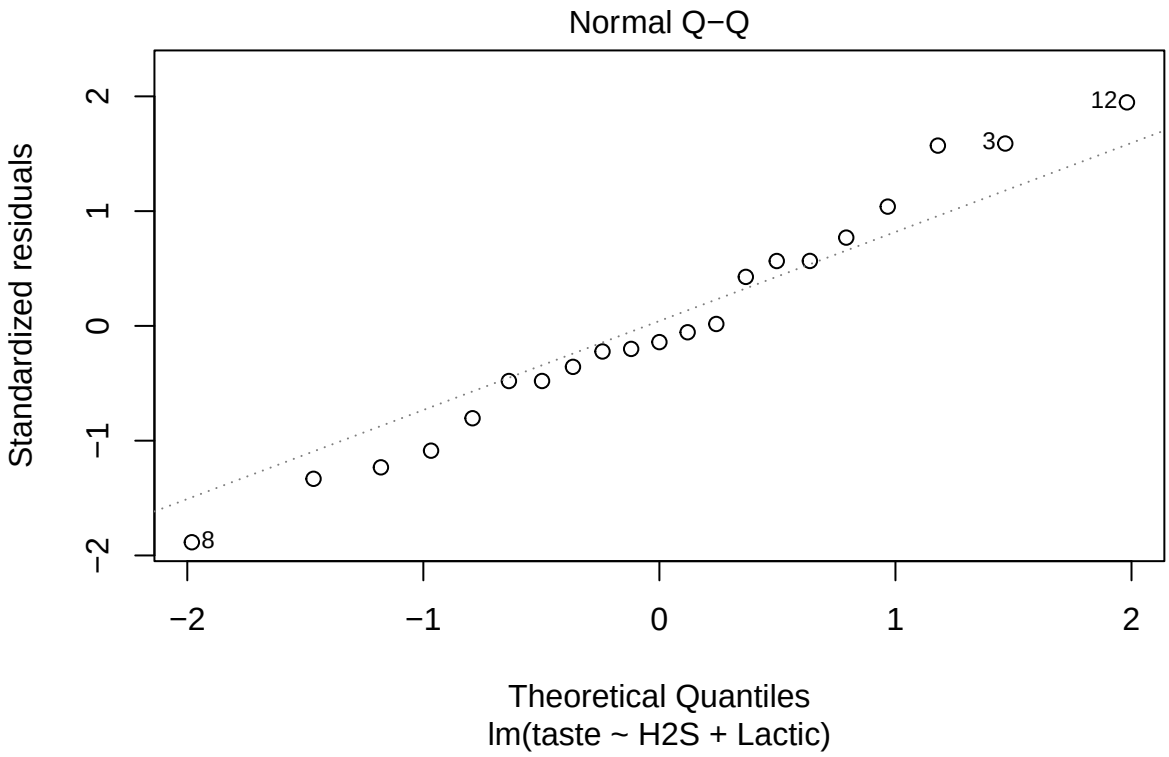

```
residuos <- resid(model.final1)
```

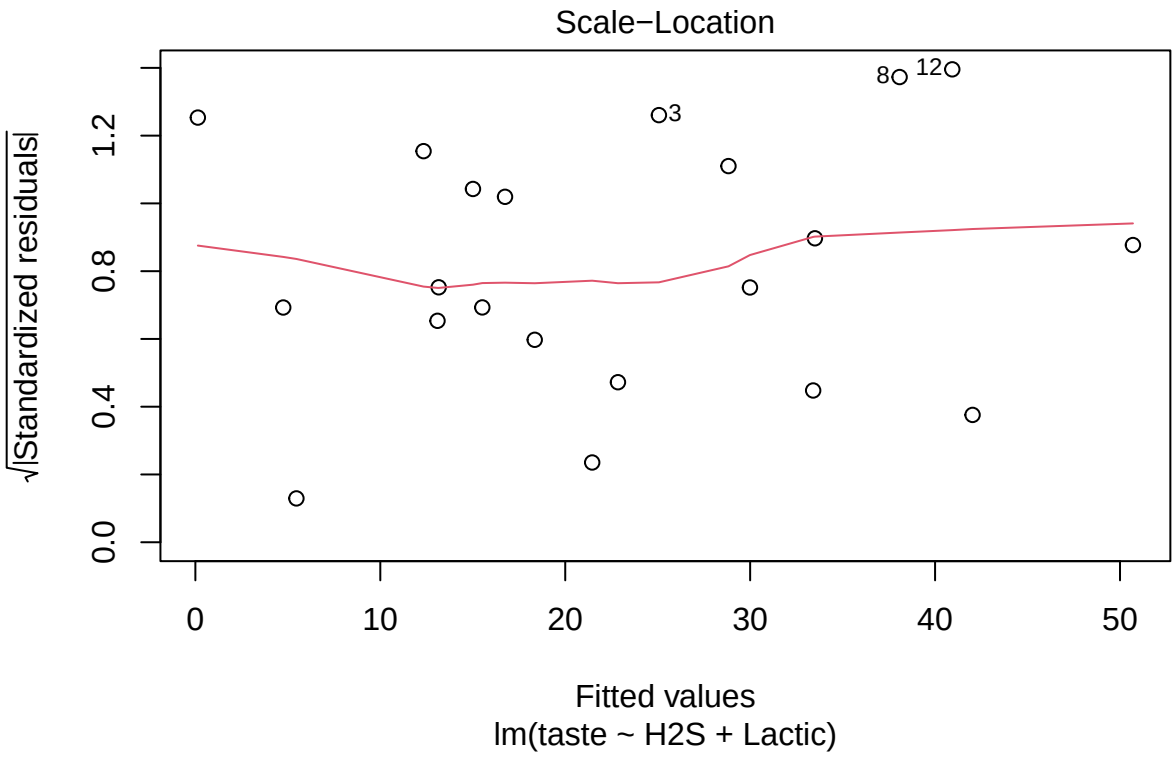
Observamos que este modelo tiene algunas características poco lineales, ya que en el primer plot hay una aparente ausencia de relación linea, lo comprobamos estadisiticamente.

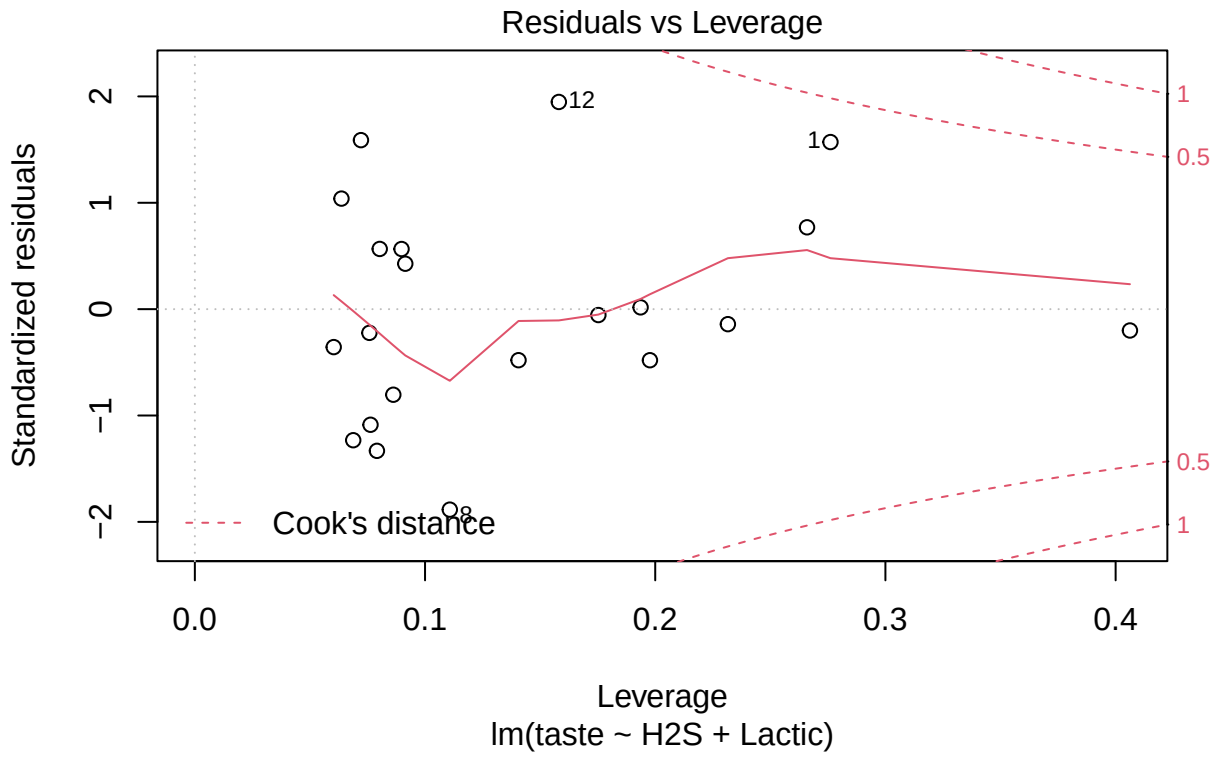
### 3.1.3 Model.final2

```
plot(model.final2)
```


Residuals vs Fitted
Residuals
Fitted values
lm(taste ~ H2S + Lactic)
20
10
0
−10
−20
0
10
20
30
40
50
12
3
8


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(taste ~ H2S + Lactic)
8
3
12
−2
−1
0
1
2


Scale−Location
√|Standardized residuals|
0.0
0.4
0.8
1.2
0
10
20
30
40
50
8
12
3
Fitted values
lm(taste ~ H2S + Lactic)




```
residuos2 <- resid(model.final2)
```

Observamos que este modelo tiene algunas características poco lineales, ya que en el segundo plot nuestros datos llegan a dispersarse considerablemente,lo comprobamos estadisiticamente.

### 3.1.4 Comprobación de hipótesis

A fin de exponer con una mayor claridad los resultados y ahorrar espacio, recogemos los resultados de los tests en una tabla en la que observamos si se verifican las hipotesis del modelo de regrsión lineal.

```
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 4.1.3
```

```
res.aov <- aov(model.final1)
res.aov2 <- aov(model.final2)
shap1 <- shapiro.test(residuos)$p.value
shap2 <- shapiro.test(residuos2)$p.value

t1 <- t.test(residuos, mu = 0, alternative = "two.sided")$p.value
t2 <- t.test(residuos2, mu = 0, alternative = "two.sided")$p.value

aov1L <- summary(res.aov)[[1]][["Pr(>F)"]][1]
aov2L <- summary(res.aov2)[[1]][["Pr(>F)"]][1]
aov2H <-  summary(res.aov2)[[1]][["Pr(>F)"]][2]

dw1 <- durbinWatsonTest(model.final1)[[3]]#CORREGIR PROBLEMILLA; NO SON ESTOS LOS P
dw2 <- durbinWatsonTest(model.final2)[[3]]
```

```
df_hip <- data.frame("Distribución_normal" = c(shap1,shap2, 0.05,"Shapiro-Wilk"),
                     "Media_0" = c(t1,t2, 0.05,"t-Test"),
                     "Varianza_no_constante_H2S" = c("No participa en modelo",aov2H,0.05,"ANOVA"),
                     "Varianza_no_constante_L" = c(aov1L,aov2L,0.05,"ANOVA"),
                     "No_Autocorrelación" = c(dw1,dw2,0.05, "Durvin-Watson"))
rownames(df_hip) <- c("model.final1","model.final2","p-valor","Test usado")
```

```
df_hip
```

```
##              Distribución_normal Media_0 Varianza_no_constante_H2S
## model.final1   0.255278648824673       1    No participa en modelo
## model.final2   0.932900725837539       1        0.0327545916369228
## p-valor                     0.05    0.05                       0.05
## Test usado          Shapiro-Wilk  t-Test                      ANOVA
##              Varianza_no_constante_L No_Autocorrelación
## model.final1    1.38792275314727e-05              0.584
## model.final2    7.36620192187723e-06              0.482
## p-valor                         0.05               0.05
## Test usado                     ANOVA      Durvin-Watson
```

A pesar de nuestras suposiciones iniciales, los modelos satisfacen todas las hipótesis de un modelo de regresión lineal.
Es interesante observar que los residuos del segundo modelo se ajustán mejor a una normal y la variación de la autocorrelación al añadir un segundo predictor.

## 3.2 *Outliers* y observaciones con leverage alto.

Anteriormente hemos tratado que nuestra muestra no contiene *outliers*. ¿Pasa lo mismo con las observaciones influyentes?

Realizamos un estudio de estos últimos a través de distintos criterios: ### Estuudio de model.final1

### 3.2.1 Criterio 1: valores leverage (hii) mayores que 2p/n

```
X <- model.matrix(~ H2S + Lactic, data = cheddar)
H <- X %*% solve(t(X) %*% X) %*% t(X)
hii <- diag(H)

hCV <- 2 * 3 / 30
sum(hii > hCV)
```

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

```
which(hii > hCV) # 6
```

```
## 6
## 6
```

### 3.2.2 Criterio 2: valores |DFFITS| son mayores que 2*sqrt(p/n)

```
dffitsCV <- 2 * sqrt(3 / 30)
dffitsmodel <- dffits(model.final1)

sum(dffitsmodel > dffitsCV)
```

```
## [1] 2
```

### 3.2.3 Criterio 3: valores |DFBETAS| mayores que 2/sqrt(n)

```
dfbetaCV <- 2 / sqrt(30)
dfbetamodel <- dfbeta(model.final1)
dfbetamodel
```

```
##    (Intercept)       Lactic
## 1  10.98453709 -6.857931612
## 2   0.04813185 -0.204187174
## 3  -0.50663219  0.739444044
## 5   1.52600073 -0.930155631
## 8   3.53325599 -2.954743648
## 9   0.50645416 -0.258946998
## 10  0.37463452 -0.499299660
## 11 -0.36774193  0.342168554
## 12 -5.72680931  4.515468710
## 13 -2.19024714  1.309997368
## 14  1.89785939 -0.955345774
## 16 -1.19914765  1.035728744
## 19  1.18937074 -1.238273837
## 22 -0.34585424  0.164146758
## 23  0.67946353 -0.123569461
## 24 -5.26050279  4.025271578
## 25  0.06574074 -0.032522575
## 26 -0.57342391  0.006350913
## 27  1.36540211 -1.203660261
## 28 -3.13839299  1.685507961
## 30 -1.99398262  1.070889972
```

```
sum(dfbetamodel[, 1] > dfbetaCV)
```

```
## [1] 9
```

```
sum(dfbetamodel[, 2] > dfbetaCV)
```

```
## [1] 7
```

```
#sum(dfbetamodel[, 3] > dfbetaCV) # Yo diria que esta se quita para evitar error

which(dfbetamodel[, 1] > dfbetaCV)
```

```
##  1  5  8  9 10 14 19 23 27
##  1  4  5  6  7 11 13 15 19
```

```
#which(dfbetamodel[, 3] > dfbetaCV) # Yo diria que esta se quita para evitar error
```

### 3.2.4 Conclusión observaciones influyentes

Una vez hemos localizado estas observaciones, en este caso las hay, las retiramos de la muestra con el fin de tener una muestra adecuada para trabjar

```
obs.out <- c(6, 7, 8, 12, 15)
cheese <- cheddar[-obs.out, ]
```

Ahora ya estamos en condiciones de realizar un estudio adecuado de las condiciones de regresión lineal.

### 3.3 Estudio de modelos en la muestra revisada.

```
model.exh <- regsubsets(taste ~ ., data = cheddar[train, ], method = "exhaustive")
summary(model.exh)
```

```
## Subset selection object
## Call: regsubsets.formula(taste ~ ., data = cheddar[train, ], method = "exhaustive")
## 3 Variables  (and intercept)
##         Forced in Forced out
## Acetic      FALSE      FALSE
## H2S         FALSE      FALSE
## Lactic      FALSE      FALSE
## 1 subsets of each size up to 3
## Selection Algorithm: exhaustive
##          Acetic H2S Lactic
## 1  ( 1 ) " "    " " "*"
## 2  ( 1 ) " "    "*" "*"
## 3  ( 1 ) "*"    "*" "*"
```

Definimos...

```
predict.regsubsets <- function(object, newdata, id, ...) {
  form <- as.formula(object$call[[2]])
  mat <- model.matrix(form, newdata)
  coefi <- coef(object, id = id)
  xvar <- names(coefi)
  mat[, xvar] %*% coefi
}
```

Calculamos...

```
val.errors <- rep(NA, 3)
Y <- cheddar[test, ]$taste
for (i in 1:3) {
  Yhat <- predict.regsubsets(model.exh, newdata = cheddar[test, ], id = i)
  val.errors[i] <- mean((Y - Yhat)^2)
}
```

```
val.errors
```

```
## [1] 250.9134 142.4993 177.7624
```

¿Qué deducimos de aquí?

```
coef(model.exh, which.min(val.errors))
```

```
## (Intercept)         H2S      Lactic
##  -31.121999    3.054151   25.210179
```

Lactic es sin duda nuestro predictor mas influyente.

Ahora minimizamos el error.

```
regfit.best <- regsubsets(taste ~ ., cheddar[-obs.out, ])
coef(regfit.best, which.min(val.errors))
```

```
## (Intercept)         H2S      Lactic
##  -28.348798    3.315346   22.020523
```

Ligero cambio

## 3.4 Test de modelos

### 3.4.1 Validacion cruzada de 1

```
n <- nrow(cheese)
k <- n # numero de grupos, como es elemento a elemento hay n

folds <- sample(x = 1:k, size = nrow(cheese), replace = FALSE)
cv.errors <- matrix(NA, k, 3, dimnames = list(NULL, paste(1:3)))
for (j in 1:k) {
  best.fit <- regsubsets(taste ~ ., data = cheese[folds != j, ]) # cogemos datos del train
  for (i in 1:3) {
    pred <- predict.regsubsets(best.fit, newdata = cheese[folds == j, ], id = i) # datos test
    cv.errors[j, i] <- mean((cheese$taste[folds == j] - pred)^2)
  }
}
mean.cv.errors <- apply(cv.errors, 2, mean)
```

Ahora recogemos nuestros términos

```
mean.cv.errors
```

```
##         1         2         3
## 120.27050  68.20906  66.91742
```

```
coef(best.fit, which.min(mean.cv.errors))
```

```
## (Intercept)      Acetic         H2S      Lactic
##   -6.449428   -6.642362    3.884110   29.898710
```

### 3.4.2 Validacion en 4 grupos, cambiar la linea de k para otro numero

```
n <- nrow(cheese)
k <- 4 # numero de grupos

folds <- sample(x = 1:k, size = nrow(cheese), replace = TRUE)
cv.errors <- matrix(NA, k, 3, dimnames = list(NULL, paste(1:3)))
for (j in 1:k) {
  best.fit <- regsubsets(taste ~ ., data = cheese[folds != j, ]) # cogemos datos del train
  for (i in 1:3) {
    pred <- predict.regsubsets(best.fit, newdata = cheese[folds == j, ], id = i) # datos test
    cv.errors[j, i] <- mean((cheese$taste[folds == j] - pred)^2)
  }
}
mean.cv.errors <- apply(cv.errors, 2, mean)
```

Comprobamos los términos

```
mean.cv.errors
```

```
##         1         2         3
## 106.09030  66.22008  64.28153
```

```
coef(best.fit, which.min(mean.cv.errors))
```

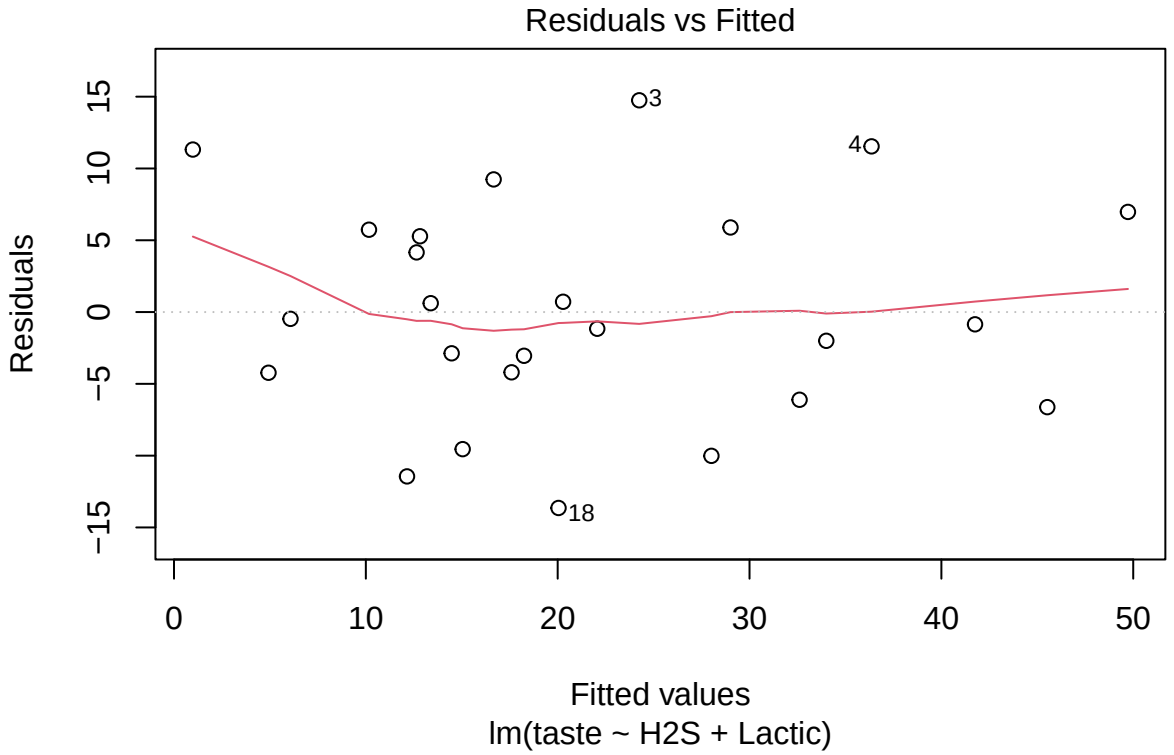
```
## (Intercept)      Acetic         H2S      Lactic
##  -13.310537   -3.386748    4.131931   20.510849
```

### 3.4.3 Comprobaciones finales

```
model.cv <- lm(taste ~ H2S + Lactic, data = cheese)
summary(model.cv)
```

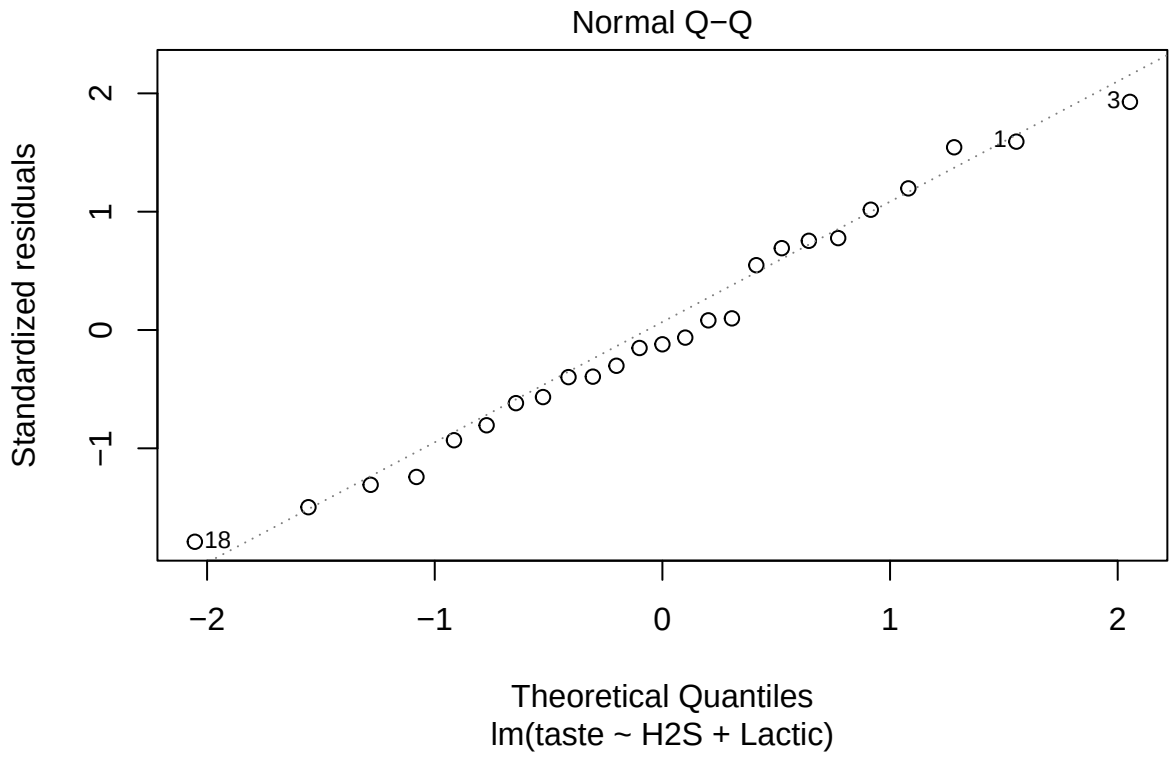
```
##
## Call:
## lm(formula = taste ~ H2S + Lactic, data = cheese)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.6439  -4.2257  -0.8544   5.7354  14.7477
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -28.349      8.042  -3.525  0.00191 **
## H2S            3.315      1.114   2.976  0.00697 **
## Lactic        22.021      7.850   2.805  0.01031 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.907 on 22 degrees of freedom
## Multiple R-squared:  0.7395, Adjusted R-squared:  0.7158
## F-statistic: 31.22 on 2 and 22 DF,  p-value: 3.751e-07
```

```
plot(lm(taste ~ H2S + Lactic, data = cheese), which = 1)
```



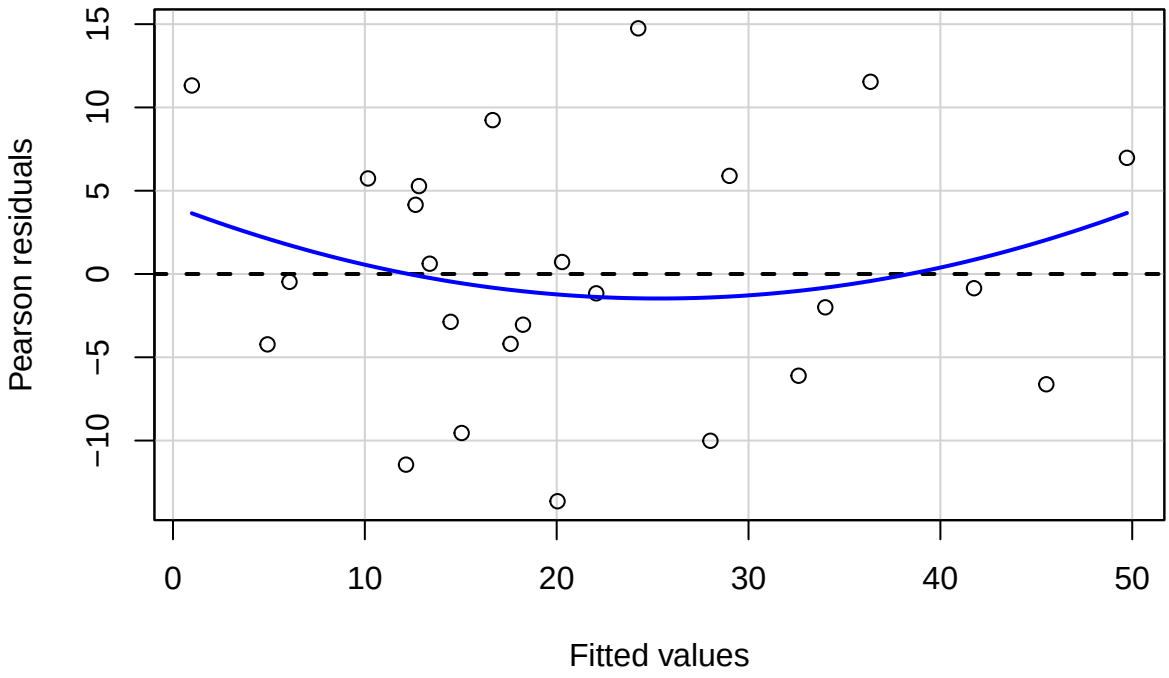

Se observa que practicamente se ajusta por completo a un modelo lineal.

```
plot(lm(taste ~ H2S + Lactic, data = cheese), which = 2)
```




Los residuos se comportan como una distribución normal de manera bastante clara

```
residualPlot(model.cv)
```




# Errores hay que moverlos aquí

## 4 Conclusión

Después de las comparaciones realizadas con respecto a nuestras comprobaciones cruzadas concluimos que el mejor modelo es **taste** ~ **Lactic**.

```
model.final <- model.final1
```

Es un modelo especialmente simple, sus coeficientes son:

```
coeff <- summary(model.final)$coeff[,1]
```

Y algo más de información, aunque ya la hemos estudiado antes la aporta

```
sumary(model.final)
```

```
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept) -39.8244    11.0385 -3.6078  0.001875
## Lactic       42.9502     7.4102  5.7961 1.388e-05
##
## n = 21, p = 2, Residual SE = 9.87699, R-Squared = 0.64
```

Una consecuencia interesante de tener solo 3 predictores es que podemos interpretar nuestro modelo de regresión como un plano en el espacio.

Primero veamos como se distribuye la variable taste en función de H2S y Lacti

```
plot_ly(x=H2S, y=Lactic, z=taste, type="scatter3d", color=taste, mode='markers') %>%
  layout(scene = list(xaxis = list(title = 'H2S (%)'),
```

```
                      yaxis = list(title = 'Lactic (%)'),
                      zaxis = list(title = 'Taste (0-100)')))
```

Y ahora podemos añadir nuestro "plano" de regresión. Marcamos en rojo las observaciones que peor se ajustan al modelo.

```
# planereg <- scatterplot3d(x=H2S, y=Lactic, z=taste, pch=16, cex.lab=1,
#                           highlight.3d=TRUE, type="h", xlab='H2S (%)',
#                         ylab='Lactic (%)', zlab='Taste (0-100)')
# plot.new(planereg$plane3d(model.final, lty.box="solid", col='mediumblue'))
# Me da error
```