Regresión: Modelos Estadísticos

Conjunto de Datos: Cheddar Faraway

Daniel López Montero Rodrigo de la Nuez Moraleda José García Rebollo
David Parro Plaza

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.

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.
- **H2S:** la concentración de sulfito de hidrógeno.
- **Lactic:** concentración de ácido láctico.

Cargamos los datos y enseñamos las primeras observaciones.

taste	Acetic	H2S	Lactic
12.3	4.543	3.135	0.86
20.9	5.159	5.043	1.53
39.0	5.366	5.438	1.57
47.9	5.759	7.496	1.81
5.6	4.663	3.807	0.99
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)**.

Pero en nuestro caso evaluamos y no hay valores *missing*.

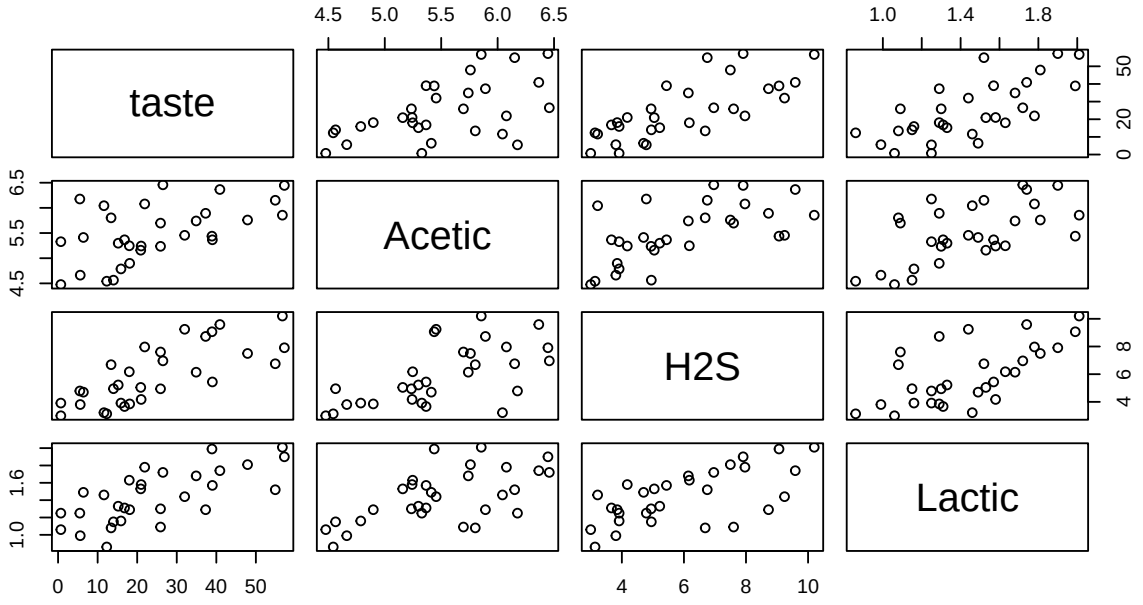
En nuestro caso, todas las variables son numéricas (**cuantitativas**). No hay ninguna variable categórica ni cualitativa. Si hubiese, tendríamos que transformarlas en variables binarias. Para ello, deberíamos hacer encoding a variables binarias, el lenguaje de programación R nos permite utilizar *as.numeric*.

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 lineal del tipo Y = f(X). También vamos a intentar **predecir** el valor de la variable Y para valores no observados de las variables X.

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 usaremos para cuantificar el error de los modelos.

Para asegurar que sea reproducible utilizamos una semilla, que permite fijar los valores pseudoaleatorios obtenidos en muchas de las funciones utilizadas.

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

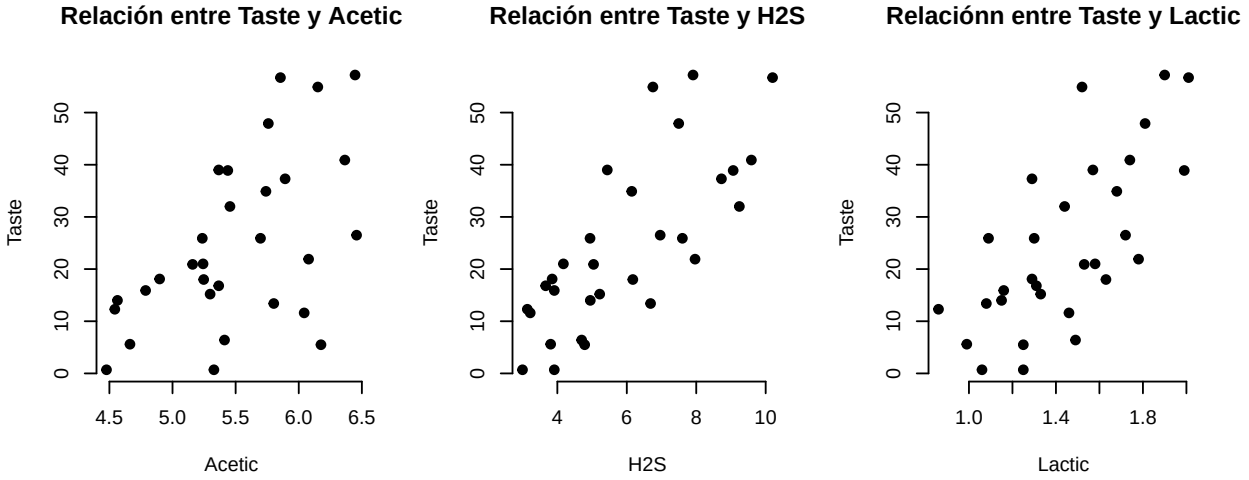


Ahora, utilizamos la función *summary* de R, la cual nos permite estimar algunos de las características de la distribución del dataset. La siguiente tabla nos muestra los estadísticos más comunes: el mínimo, máximo, mediana, media y el 1er y 3er cuartil.

Presentamos el modelo:

column	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
taste	30	24.533333	16.255382	20.950	23.345833	10.2000	0.700	57.200	56.500	0.5476046	2.385406	2.9678133
Acetic	30	5.498033	0.5708784	5.425	5.503667	0.4035	4.477	6.458	1.981	-0.0954829	2.168010	0.1042277
H2S	30	5.941767	2.1268792	5.329	5.828083	1.5005	2.996	10.199	7.203	0.4096381	1.980205	0.3883132
Lactic	30	1.442000	0.3034900	1.450	1.435417	0.2150	0.860	2.010	1.150	0.1192733	2.174947	0.0554094

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



Podemos observar que la que aperentemente guarda una menor relación lineal con taste es la variable Acetic, este hecho será reafirmado más adelante.

2 Estudio y evaluación del modelo completo.

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, β_0 y los β_j son los parámetros $j = 1, ..., p-1$, x_{ij} son los elementos de la matriz de las variables explicativas ϵ_i es el término del error aleatorio que suponemos que se distribuye como una $\mathcal{N}(0, \sigma^2)$, donde σ^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 ϵ. 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.

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

$$\hat{\beta}_0 = -28.8767696, \quad \hat{\beta}_1 = 0.3277413, \quad \hat{\beta}_2 = 3.911841, \quad \hat{\beta}_3 = 19.6705434$$

2.2 Resolución usando librerias de R

Podemos utilizar la función *lm*, ya programada en R. Definimos el modelo completo, y comprobamos las betas:

$$taste \sim Acetic + H2S + Lactic,\ data = cheddar$$

	Intercept	Acetic	H2S	Lactic
Coeficientes	-28.87677	0.3277413	3.911841	19.67054

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.

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

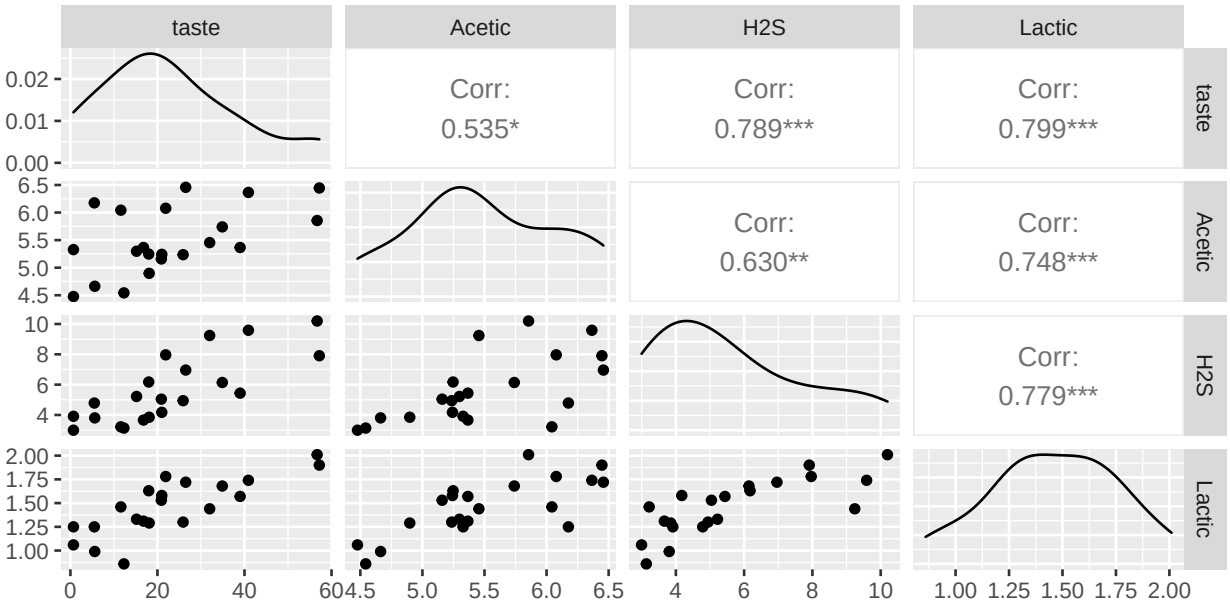
Otro aspecto que debemos saber es si la media de los errores será nula o no, para ello usamos una función implementada en R que nos afirma que en nuestro caso esto es así. Respecto a los errores tenemos que saber que se distribuyen con varianza constante, que de manera análoga comprobamos, teniendo p-valores lo suficientemente bajos como para rechazar varianza no constante. De igual manera tenemos que no hay autocorrelación en el modelo completo.

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.

term	estimate	std.error	statistic	p.value
(Intercept)	-28.8767696	19.735418	-1.4631952	0.1553991
Acetic	0.3277413	4.459757	0.0734886	0.9419798
H2S	3.9118411	1.248430	3.1334077	0.0042471
Lactic	19.6705434	8.629055	2.2795710	0.0310795

Tras realizar el summary del modelo, una observación muy importante que sacamos es que el p-valor de Acetic es muy grande (0.94) por lo que no va a ser una variable significativa y por tanto con casi toda seguridad no tiene impacto real en el modelo.

Por el contrario el resto de variables sí son significativas por tener p-valor inferior a 0.5, destacando Lactic por tener un beta mayor que H2S, lo que ppuede hacernos llegar a pensar que cobre más importancia en un modelo final. #### 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.



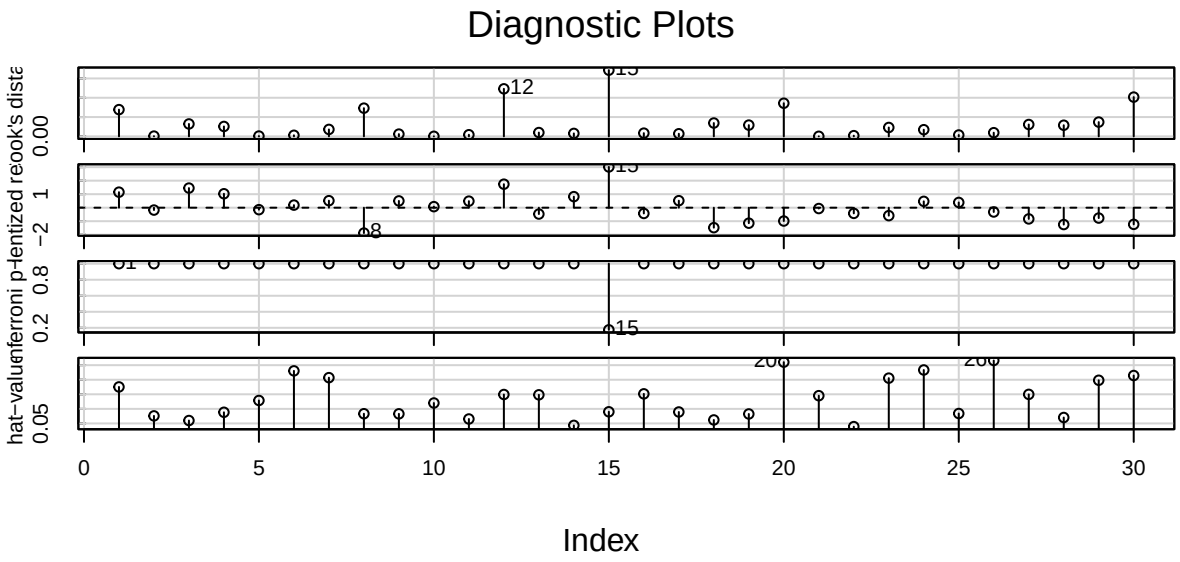
Pdemos apreciar que la variable con menor coefeciente de correlación de Pearson es Acetic,mientra que el resto de variables tienen un valor bastante mayor. Cabe destacar que todos tienen un signo positivo, lo que nos dice que en general al aumentar una aumenta la otra.

2.3 ¿Tiene *outliers* nuestra muestra?

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

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 15, que tiene un valor **Bonferroni p** de 0.17453 (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 observación 15 y por tanto deberíamos considerar si es outlier. Hemos comprobado que sucedería al eliminar esa observación de la muestra, y si bien mejoraba algún p-valor de las hipótesis de regresión, en otros los empeoraba, así que hemos decidido no eliminarla y esperarnos a ver el caso en las seeds que estudiaremos con más detalles si sale de la muestra como observación influyente.

3 ¿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.

3.0.1 Separacion del dataset en conjuntos de entrenamiento y test (70-30%)

Hemos escogido distintas semillas para estar en condiciones de realizar un estudio más amplio, en la elección de las mismas se ha intentado evitar aquellas que generaban muestras demasiado similares. Las semillas usadas son 1, 1100 y 5 posteriormente se introducirán dos más para afinar en el cálculo de errores.

Consideremos los conjuntos de entrenamiento resultantes de las semillas: *train.1*(semilla 1), *train.2*(semilla 1100), *train.3*(semilla 5).Veaamos que modelos debemos considerar en base a nuestros train.

3.0.2 Método Backward

Partimos del modelo completo estudiado en la sección anterior, eso sí, evaluado en nuestros respectivos train y aplicamos con $\alpha = 0.05$, el método de Backward, que consiste en eliminar la variable que *menos influya* a la predicción. Primero realizamos una iteración explícita del método sobre la primera semilla, y posteriormente se construyen a través de la libreria *mixlm* de R.

	Acetic	H2S	Lactic
p-valor	0.33627	0.04318	0.02365

Eliminamos Acetic del modelo debido que su p-valor es >0.05.

	H2S	Lactic
p-valor	0.05122	0.03275

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

	Lactic
p-valor	1.39e-05

El p-valor es menor que $\alpha = 0.05$, por lo que 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: **taste ~ Lactic, data = cheddar[train.1,]**.

Por otro lado los modelos backward resultantes por *mixlm* son:

taste ~ H2S + Lactic, data = cheddar[train.2,] y por otro lado **taste ~ H2S + Lactic, data = cheddar[train.3,]**.

3.0.3 Método Forward

El método Forward consiste en empezar con un modelo de una variable y vamos añadiendo las que más influyan, desarrollaremos el primer modelo de forma explícita y el resto los generaremos con *mixlm*. De esta manera tenemos:

	Acetic	H2S	Lactic
p-valor	0.0125179	2.12e-05	1.39e-05

Actualizamos añadiendo Lactic por tener el menor p-valor.

	Acetic	H2S
p-valor	0.5039877	0.0512171

Con nivel de significación $\alpha = 0.05$ este sería nuestro modelo final. Modelo resultante = **taste ~ Lactic, data = cheddar[train.1,]**
Por otro lado los modelos forward resultantes por *mixlm* son:

taste ~ H2S + Lactic, data = cheddar[train.2,] y por otro lado **taste ~ H2S + Lactic, data = cheddar[train.3,]**.

3.1 Construcción por criterios

En esta subsección trataremos de encontrar un candidato a mejor modelo, construyendo nuestros modelos usando distintos enfoques. Tras aplicar los siguientes criterios a la hora del desarrollo de modelos: R^2 ajustado, Cp de Mallows, Criterio de Informacion de Bayes (BIC), Criterio de Informacion de Akaike (AIC) eligiendo el que nos de un menor o mayor coeficiente según el método (los desarrolos se pueden encontrar en el script), llegamos a las siguientes conclusiones:
solo aparece un modelo nuevo usando el criterio del estadístico R^2 para la primera semilla, **taste ~ H2S + Lactic, data = cheddar[train.1,]**.
Notamos que la combinación de H2S + L aparece en todos nuestros conjuntos de entrenamiento en algún momento, es candidata a ser nuestra mejor elección.

Comparamos los modelos obtenidos hasta ahora en su respectiva muestra de entrenamiento con el modelo completo en ese conjunto de entrenamiento.

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
19	1853.543	NA	NA	NA	NA
17	1410.701	2	442.8415	2.668285	0.0982145

	train1: L vs Completo	train1: H2S + L vs Completo	train2: H2S + L vs Completo	train1: H2S+ L vs Completo
p-valor	0.0982145	0.3362689	0.9551123	0.5551292

Con estos p-valores podemos decir que con un nivel de significación α ningún modelo es notablemente diferente de su contraparte salvo en el caso de los modelos resultantes en *train2* esto puede ser por la cantidad de observaciones influyentes presentes en la muestra, lo trataremos en la siguiente sección.

4 Diagnostico: Comprobaciones de hipotesis, outliers y observaciones influyentes

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

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

4.1 ¿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 significación α adecuado:

1. Los errores ϵ_i tienen distribución normal.

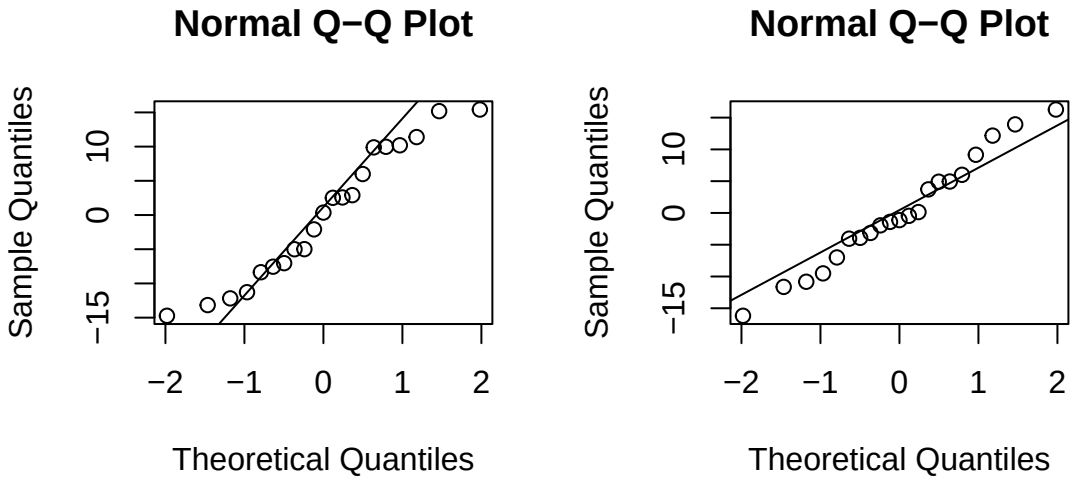
2. Los errores ϵ_i tienen media cero.

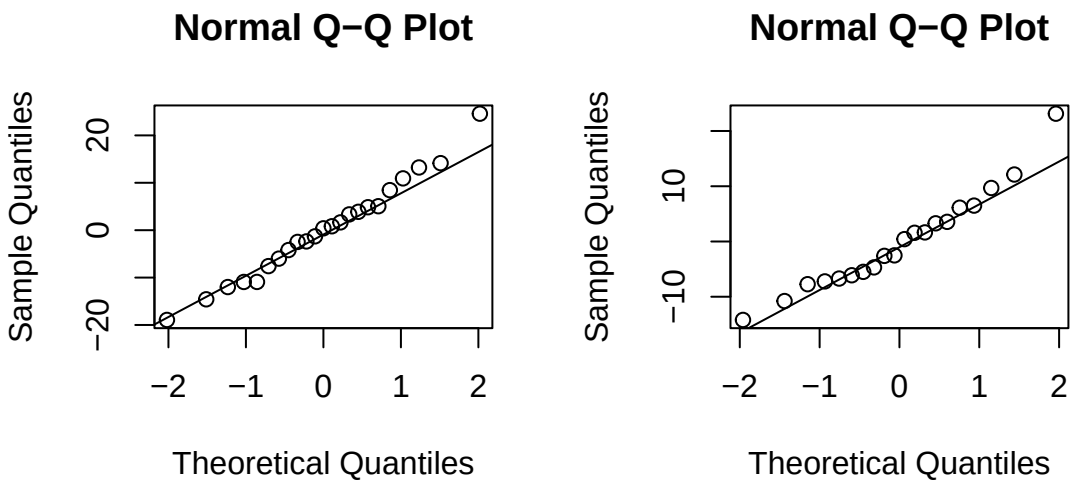
3. Los errores ϵ_i tienen varianza constante.

4. Los errores ϵ_i no están correlacionados.

	train1: L	train1: H2S + L	train2: H2S + L	train3: H2S + L	Nivel de siginificación	Test utilizado
Linealidad	0.14	0.609	0.812	0.816	0.05	Resettest
Normalidad	0.255	0.933	0.982	0.452	0.05	Test Shapiro Wilk
Media = 0	1	1	1	1	0.05	t-test
Varianza constante	0.63	0.438	0.392	0.483	0.05	Test ncv
Correlación	0.608	0.508	0.924	0.958	0.05	Test de Durbin-Watson

Podemos observar que se verifican a nivel de significación $\alpha = 0.05$ se verifican todas las hipótesis de modelo de regresión lineal. En las siguientes gráficas podemos observar como los residuos de los modelos **taste ~ H2S + L** de los conjuntos *train2* y *train3* se comportan mejor que cualquiera de los modelos propuestos en la muestra *train1*





Posteriormente, realizamos un estudio de la colinealidad. Para ello, observamos que el modelo **taste ~ Lactic , data=cheddar[train.1,]** solo tiene un predictor luego no hay presente ningún tipo colinealidad.

	H2S	Lactic
train1: L	Solo tiene un predictor	Solo tiene un predictor
train1: H2S + L	2.54626778688363	2.54626778688363
train2: H2S + L	1.79449729956943	1.79449729956943
train3: H2S + L	1.73307700400943	1.73307700400943

Nuestros valores son muy buenos, entendiendo por bueno VIF < 10, por lo tanto no tenenemos que preocuparnos de una colinealidad grave entre las variables.

4.2 Estudio de outliers

A fin de obtener distintos puntos de vista utilizaremos dos métodos, un valor de Bonferroni en un estadístico $t_{1-\frac{\alpha}{2n};n-p-1}$ y la función *outlierTest* que utiliza p-valores de Bonferroni obtenidos a través de t-tests.

En los cuatro modelos y bajo los dos criterios no se obtiene ninguna observación que se pueda aceptar como *outlier* a nivel de significación $\alpha = 0.05$

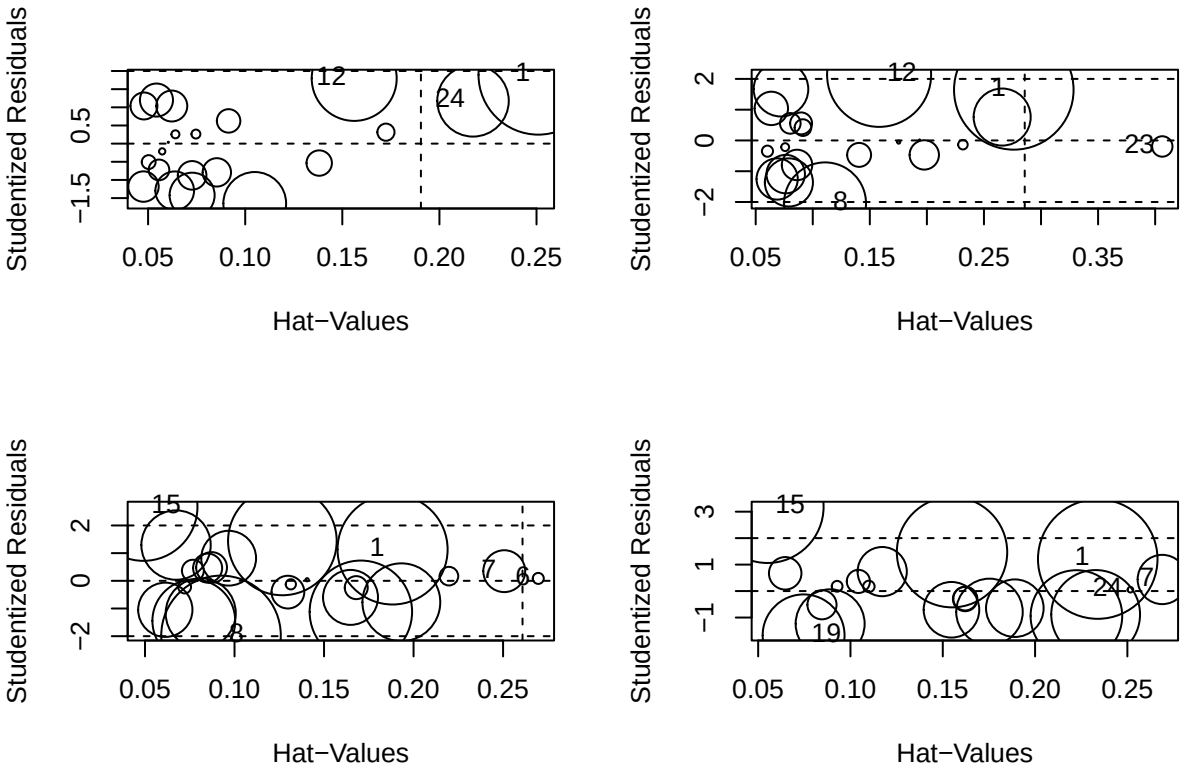
4.3 Estudio de observaciones Influyentes

Al igual que en la anterior sección a fin de poder encontrar el criterio que mejor se ajuste a nuestro modelo, en el que hay que tener en cuenta que es una muestra de tamaño reducido, utilizaremos varios criterios, además de distintas técnicas gráficas. Los criterios usados serán:
1.Criterio 1: valores leverage (hii) mayores que $\frac{2p}{n}$.
2.Criterio 2: valores |DFFITS| son mayores que $2 \cdot \sqrt{\frac{2p}{n}}$.
3.Criterio 3: valores |DFBETAS| mayores que $2 \cdot \sqrt{\frac{2p}{n}}$.
4.Criterio 4: *InfluencePlot*.

En la siguiente tabla observamos las observaciones influyentes obtenidas por cada criterio.

	Criterio 1	Criterio 2	Criterio 3	Criterio 4
train1: L	1,5,16,23,24,26	1,12,24	1,3,5,8,9,12,13,14,16,19,23,24,27,28,30	1,12,24
train1: H2S + L	23	12	1,3,8,9,11,12,13,14,19,23,24,26,27,28,30	1,8,12,23
train2: H2S + L	6	Ninguno	1,3,4,7,8,9,11,12,16,17,20,29,30	1,6,7,8,15
train3: H2S + L	Ninguno	Ninguno	1,4,5,7,12,14,16,17,19,23,27,29	1,7,15,19,24

Tomamos la decisión de elegir los elementos que se repitan en varios ya que si eliminásemos los del criterio 3 el resultado sería demasiado pequeño para ser apto para la regresión lineal. Nuestro criterio a seguir es que estén en al menos dos criterios de los restantes.



Estas observaciones son las presentes en la cuarta columna de la tabla y en general son en las primeras en las que nos fijemos a la hora de ver si las tomamos como influyentes o no.

Bajo nuestro criterio quedarían como posibles influyentes las siguientes observaciones:

	Observaciones influyentes
train1: L	1,12
train1: H2S + L	1,12,23
train2: H2S + L	2,6
train3: H2S + L	No se repite ninguno

Ahora es cuando tenemos que determinar si van a ser influyentes o no, esto lo veremos comparando los

modelos con las muestras de entrenamiento sin eliminar influyentes y con las muestras modificadas, y evaluar si mejoran o empeoran los p-valores de las hipótesis de los modelos de regresión.

4.3.1 taste ~ + Lactic, data = cheddar[train.1,] vs taste ~ Lactic, data = cheddar[train.1inf,]

	train1: L	train1: L, sin influyentes	Test utilizado
Linealidad	0.14	0.558	Resettest
Normalidad	0.255	0.933	Test Shapiro Wilk
Media = 0	1	1	t-test
Varianza constante	0.63	0.357	Test ncv
Correlación	0.562	0.664	Test de Durbin-Watson

Nuestras hipótesis del modelo lineal se ven notablemente mejoradas, tenemos una distribución más parecida a una normal, mejor distribuida en torno a una lineal.
Realizamos un pequeño intercambio en el que nuestra varianza parece menos constante pero se mantiene por encima de todos los niveles de significación habituales.
Tomamos la decisión de actualizar *cheddar[train.1,]* con *cheddar[traininf.1,]*, que es la versión sin observaciones influyentes. ### taste ~ H2S + Lactic, data = cheddar[train.1,] vs taste ~ H2S + Lactic, data = cheddar[train.1critinf,]

	train1: H2S + L	train1: H2S + L, sin influyentes	Test utilizado
Linealidad	0.609	0.358	Resettest
Normalidad	0.933	1	Test Shapiro Wilk
Media = 0	1	1	t-test
Varianza constante	0.439	0.376	Test ncv
Correlación	0.476	0.772	Test de Durbin-Watson

Este es un caso un tanto particular, tenemos unos datos perfectamente distribuidos, pero ya lo estaban antes practicamente. A cambio de eso perdemos un cierto grado de linealidad que hay que tener en cuenta, en este caso no hay una diferencia tan notable con el modelo como en que solo incorporaba a Lactic.

Tomamos la decisión de actualizar *cheddar[train.1crit,]* con *cheddar[train.1critinf,]*, que es la versión sin observaciones influyentes.

4.3.2 taste ~ H2S + Lactic, data = cheddar[train.2,] vs taste ~ H2S + Lactic, data = cheddar[train.2inf,]

	train2: H2S + L	train2: H2S + L, sin influyentes	Test utilizado
Linealidad	0.812	0.83	Resettest
Normalidad	0.982	0.984	Test Shapiro Wilk
Media = 0	1	1	t-test
Varianza constante	0.392	0.419	Test ncv
Correlación	0.928	0.966	Test de Durbin-Watson

En cierta manera es notable que al eliminar nuestras influyentes y en un *dataset* tan pequeño no genere a penas diferencia, en todo caso nos mejora todas las hipótesis del modelo lineal, se puede afirmar que es un modelo mejor para trabajar sobre el a priori.

Tomamos la decisión de actualizar *cheddar[train.2,]* con *cheddar[traininf.2,]*, que es la versión sin observaciones influyentes.

4.3.3 taste ~ H2S + Lactic, data = cheddar[train.3,]

En este modelo, bajo el criterio de elección de observaciones influyentes que elegimos para tratar nuestros datos, este modelo no presentaba ninguna observación influyente, por lo que se mantiene como está.

5 Errores de Test. Comparacion de Modelos

Nuestro razonamiento para enfrentarnos a esta sección es el siguiente, nos han salido dos modelos posibles y para comprender cual se ajusta mejor a nuestros datos vamos a escoger cinco seeds y evaluar cada modelo en todas ellas, a fin de hacer una media de los errores.

Para esto usamos las tres seeds que hemos utilizado a lo largo del documento, y le añadimos otras dos elegidas al azar. Nótese que evaluar en la primera seed ya está hecho, pues en esta nos salían los dos modelos a considerar. Además en las seeds dos y tres ya está hecho para el modelo con H2S + Lactic, pero hay que repetir el proceso para el otro modelo. De esta manera realizamos los mismos cálculos que los realizados en la parte de diagnóstico, sobre las combinaciones. Esto es, nos aseguramos que cada modelo con todos los train cumpla las hipótesis de normalidad, media de errores nula, homocedasticidad, linealidad y autocorrelación. Después de esto nos planteamos si tiene datos influyentes según las funciones adecuadas, en caso de tenerlos realizamos una prueba. los quitamos del train y vemos si se mejora el p-valor de alguna de las características anteriores y en base a eso decidimos si eliminamos las observaciones influyentes o no.

Con todo eso realizado llegamos a la siguiente tabla, que nos permite asumir como validos todos los casos y calcular sus errores.

	Distribución_normal	Media_0	Varianza_no_constante	No_Autocorrelación
S1 taste ~ H2s + Lactic	0.9997	1.00	0.3764	0.748
S1 taste ~ Lactic	0.2879	1.00	0.3574	0.642
S2 taste ~ H2s + Lactic	0.9840	1.00	0.4193	0.962
S2 taste ~ Lactic	0.8103	1.00	0.8584	0.860
S3 taste ~ H2s + Lactic	0.4522	1.00	0.4832	0.978
S3 taste ~ Lactic	0.7410	1.00	0.6734	0.596
S4 taste ~ H2s + Lactic	0.4865	1.00	0.9978	0.746
S4 taste ~ Lactic	0.5462	1.00	0.5620	0.996
S5 taste ~ H2s + Lactic	0.9353	1.00	0.4861	0.182
S5 taste ~ Lactic	0.9132	1.00	0.8854	0.098
Nivel de significacion	0.0500	0.05	0.0500	0.050

Calculamos los errores de cada método como la media de los errores del método aplicado en cada seed. De esta manera, el error medio obtenido es:

	Modelo H2S+Lactic	Lactic
errores	11.62746	96.97775

Como el error del modelo H2S + Lactic es menor, ese es el modelo que llamaremos final y que se ajustará lo mejor posible a los datos de nuestro fichero.

6 Conclusión: presentación del modelo final

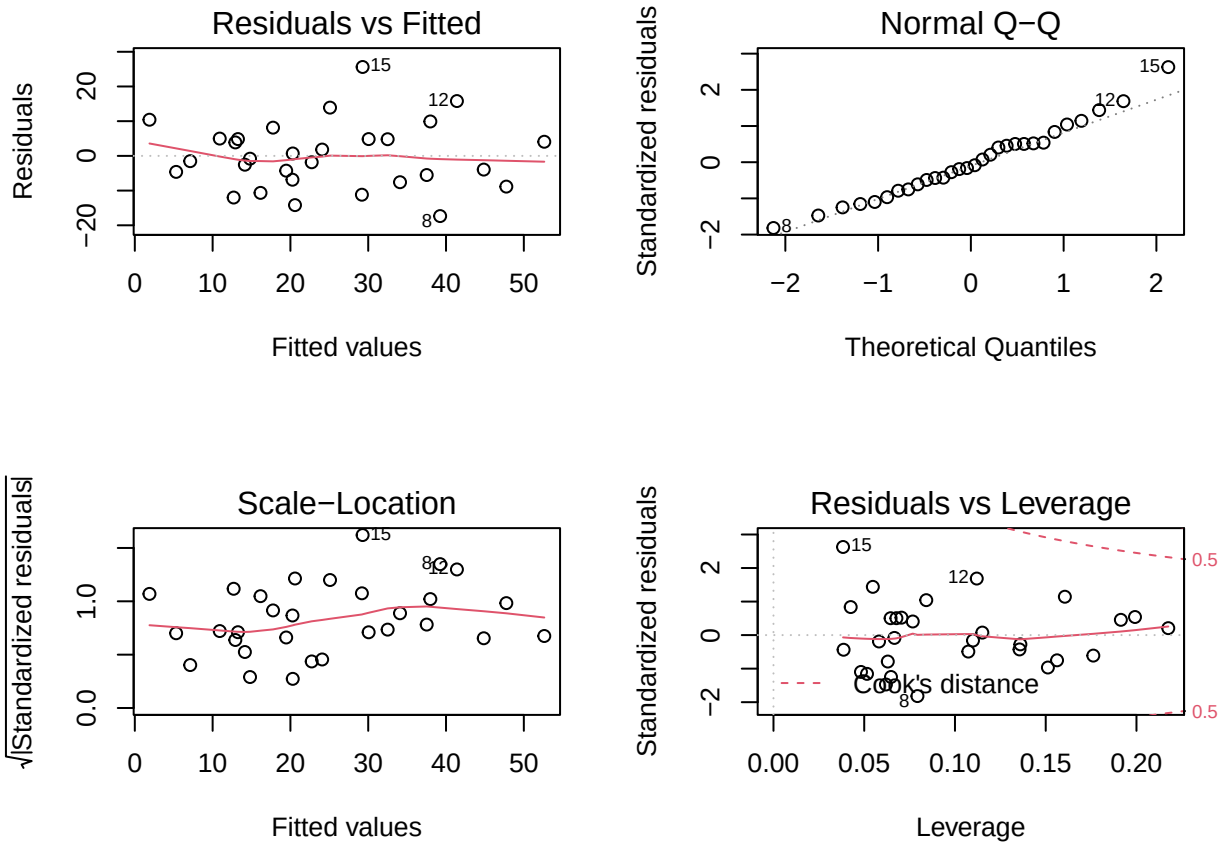
Finalmente por comparación de los errores sabemos que el modelo final que presentamos es: taste ~ H2S + Lactic , que tiene sentido ya que Acetic no era significativa. otro hecho que lo reafirma es que en la

construcción de modelos salieron en todos los casos este modelo, salvo en el método backward y en el método forward de la primera semilla.

term	estimate	std.error	statistic	p.value
(Intercept)	-27.591815	8.981825	-3.071961	0.0048129
H2S	3.946267	1.135692	3.474769	0.0017429
Lactic	19.887204	7.959009	2.498704	0.0188499

Presentamos el summary de nuestro modelo, del que sacaremos más informción en un futuro.

A su vez enseñamos los graficos de este modelo:



Ahora vamos a asegurarnos que verifica las hipótesis para una regresión lineal y además de revisar con outlierTest que no tenemos outliers. También planteamos una hipótesis sobre la nulidad de la media de los errores y vemos que su varianza es constante además de calcularla.

	p-Outliers	p-Normalidad	p-Media nula	p-Homocedasticidad	Valor de varianza	p-Autocorrelación	p-Linealidad
model.y	0.0060495	0.8106746	1	0.2770139	9.942362	0.2	0.9497967

Dicho esto procedemos a la presentación del modelo con sus betas asociados, que si bien se pueden recoger del summary, también los calculamos de forma matricial igual que se hizo con el modelo completo en su momento.

El resultado sería **taste ~ -27.591815 +3.9946267 H2S +19.887204 Lactic** Nótese que sigue siendo acorde al modelo completo donde el β de Lactic es muy superior en comparación al de H2S.

Presentamos ahora R^2 y R^2_{adj}, estas como antes se pueden recoger directamente del summary, también las calculamos a partir de los errores y la tabla anova, dando el resultado de 0.6517024 y 0.6259025 respectivamente. Además presentamos el vector de p-valores.

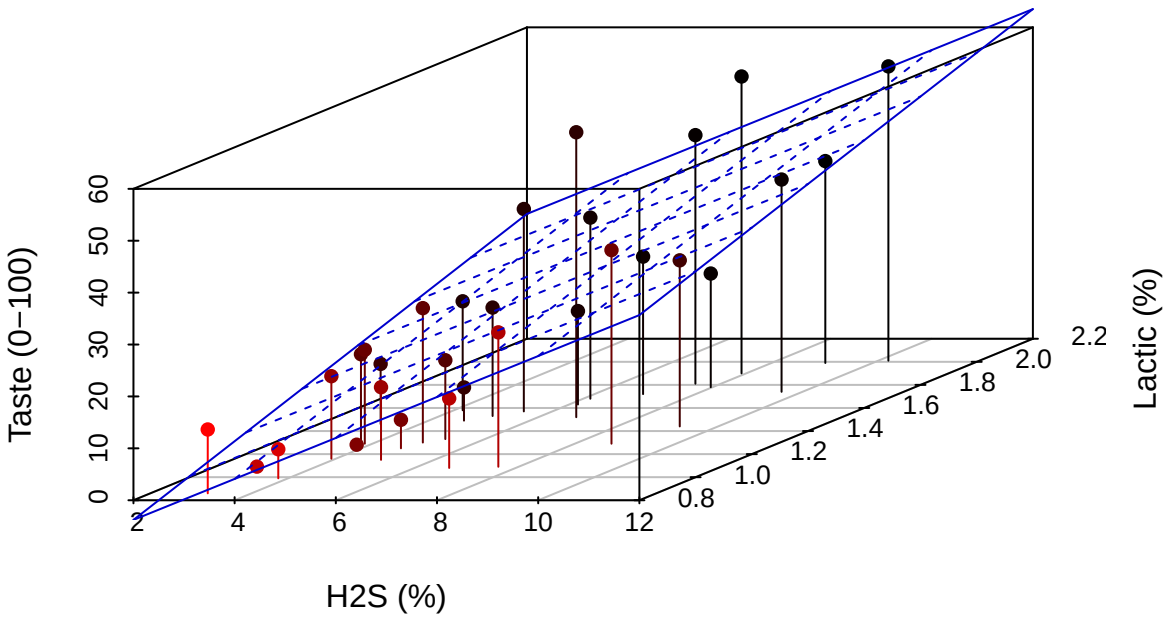
	Intercept	H2S	Lactic
P-valores	0.0048129	0.0017429	0.0188499

Observamos que todos los p-valores están por debajo de nuestra alpha de referencia.

También presentamos intervalos para las betas de Bonferroni y Scheffé:

	Bonferroni	Bonferroni	Scheffé	Scheffé
H2S	1.39408948776095	6.49844518328319	0.958193950963627	6.93434072008051
Lactic	2.00136600296324	37.7730429494495	-1.0534199984135	40.8278289508262
Porcentaje	5%	95%	5%	95%

Finalmente buscamos una representación de la regresión que tenemos en 3 dimensiones ya que el modelo final consta de una variable respuesta y dos predictoras y vemos el plano de regresión, marcando en rojo las observaciones que peor se ajustan.



7 Anexo

```
#
# install.packages("faraway")
# install.packages("leaps")
# install.packages("MASS")
# install.packages("PASWR")
# install.packages("car")
# install.packages("ggplot2")
# install.packages("GGally")
# install.packages("corrplot")
# install.packages("scatterplot3d")
# install.packages("lmtest")
# install.packages("plotly")
# install.packages("mixlm")
#
#
# library(faraway)
# library(leaps)
# library(MASS)
# library(PASWR)
# library(car)
# library(ggplot2)
# library(GGally)
# library(corrplot)
# library(plotly)
# library(scatterplot3d)
# library(lmtest)
# library(mixlm)
#
#
#
#
#
# # 1) Introduccion
#
# data(cheddar)
# attach(cheddar)
#
# # Variable Respuesta: taste
# # Variables Predictoras: Acetic, H2S, Lactic
#
#
# # Estudiamos el tipo de las variables que van a formar parte de los posibles modelos
# sapply(cheddar, class) # todas las variables son numericas
# head(cheddar)
#
# # Comprobamos que no hay entradas vacias
# any(is.na(cheddar))
#
# # De haberlas habido, podriamos haber tomado las siguientes decisiones:
# #  - eliminar las observaciones con valores NA
# #  - eliminar variables si muchas de las entradas vacias aparecen en ella
```

```
# #  - emplear métodos para predecir que valores deberian apercer en dichas entradas
#
#
#
# # Histogramas de todas las variables
# ids <- names(cheddar)
# layout(matrix(1:4, nrow = 1))
# y_lab_string <- "Cantidad"
# for (id in ids) {
#   hist(cheddar[, id], xlab = id, ylab = y_lab_string, main = paste("Histogram of ", id))
#   y_lab_string <- ""
# }
#
#
# # Graficas de las relaciones entre variables.
# plot(cheddar)
#
# # Graficas de dispersion entre la variable respuesta "taste" y las variables predictoras.
# layout(matrix(1:3, nrow = 1))
#
# plot(Acetic, taste,
#      main = "Relación entre Taste y Acetic",
#      xlab = "Acetic", ylab = "Taste",
#      pch = 19, frame = FALSE)
#
#
# plot(H2S, taste,
#      main = "Relación entre Taste y H2S",
#      xlab = "H2S", ylab = "Taste",
#      pch = 19, frame = FALSE)
#
#
# plot(Lactic, taste,
#      main = "Relación entre Taste y Lactic",
#      xlab = "Lactic", ylab = "Taste",
#      pch = 19, frame = FALSE)
#
#
# layout(matrix(1:1, nrow = 1))
# summary(cheddar)
#
#
#
#
#
#
# # 2) Estudio y evaluacion del modelo completo
#
# x <- model.matrix( ~ Acetic + H2S + Lactic, data = cheddar)
# betahat <- solve(crossprod(x, x), crossprod(x, taste))
# betahat <- c(betahat)
# betahat
#
# # Comprobamos el resultado con funciones ya implementadas
```

```
# model.all <- lm(taste ~ ., data = cheddar)
# summary(model.all)
# model.all$coefficients
#
# anova(model.all)
#
#
#
# # Intervalos de confianza de las betas del modelo completo
# confint(model.all)
#
# # Observamos que 0 esta en el intervalo de confianza de beta_0 y beta_Acetic, planteamos dos tests
# #    con hipótesis nula beta_i = 0 y hipotesis alternativa beta_i != 0.
#
# # Comenzamos con beta_Acetic pues el valor estimado es más cercano a 0 que el de beta_0
# modnoAcetic <- lm(taste ~ H2S + Lactic, data=cheddar)
# anova(modnoAcetic,model.all) # no solo el p-valor > 0.05 sino que de hecho p-valor ~ 1, aceptamos la
#
# modnoInterceptor <- lm(taste ~ H2S + Lactic + 0, data=cheddar) # modelo reducido sin interceptor
# anova(modnoInterceptor,model.all) # el p-valor es menor que 0.05, luego no es suficiente para rechaza
#
#
# # Veamos los p-valores de las variables predictoras
# summary(model.all)$coeff[,4]
# # Observamso que el p-valor de Acetic es el mas elevado, este resultado junto con el contrate de hipo
# #  podemos deducir que la variable Acetic no sera muy relevante para el estudio de los distintos mode
#
#
# # Correlaciones de las variables que intervienen en el modelo completo
# cor(cheddar)
# ggpairs(cheddar)
#
# mat_cor <- cor(cheddar, method = "pearson")
# corrplot(mat_cor, type = "upper", order = "hclust", tl.col = "black", tl.srt = 45)
#
#
# # Por ultimo veamos si hay outliers en el modelo completo
# Model.all<-fortify(model.all)
# outlierTest(model.all) # no los hay
#
# # Por el metodo de Bonferroni
# alpha <- 0.05
# BCV <- qt(1-alpha/(2*30),26) # el valor critico de Bonferroni t_{1-alpha/2n;n-p-1}, n=30,p=3
# BCV
# sum(abs(rstudent(model.all))>BCV) # el metodo no nos da ninguna observacion
#
# sort(abs(rstandard(model.all)),decreasing = TRUE)[1:3] # estos valores estan relativamente lejanos de
#
#
# # Veamos si por influenceIndexPlot() hay alguna observacion que pueda ser relevante
# influenceIndexPlot(model.all) # en la tercera grafica se ve que la observacion 15 tiene un comportami
#
# # Comprobemos si los supuestos de los modelos de regresion lineal mejoran al retirar esta observacion
```

```
# #  de nuestro dataset. De ser asi, aceptamos que pueda ser una observacion que pejudique a nuestro mo
# model.all_15<- lm(taste ~ ., data = cheddar[-15,])
#
# # Homocedasticidad
# ncvTest(model.all)
# ncvTest(model.all_15) # empeora
#
# # Normalidad
# shapiro.test(resid(model.all))
# shapiro.test(resid(model.all_15))# mejora
#
# # Autocorrelacion
# durbinWatsonTest(model.all)
# durbinWatsonTest(model.all_15)# mejora
#
# # Linealidad
# resettest(model.all, power=2:3, type="regressor", data=cheddar)
# resettest(model.all_15, power=2:3, type="regressor", data=cheddar)# empeora
#
# # En el computo global no parece que los p-valores de estos tests den a entender una mejoria general
# #  Si bien es cierto, tampoco los empeora y dependiendo del criterio a seguir podriamos decidir si es
# #  Nosotros hemos decidido no considerar la observacion 15 como tal.
#
#
#
#
#
# # 3) Seleccion del mejor modelo. Metodos por pasos y por criterios
#
# # Separacion del dataset en conjuntos de entrenamiento y test (70-30%)
#
# # Hemos considerado varias semillas para abarcar más modelos. Además, hemos intentado evitar
# #  en la medida de lo poible que se repitan muchos elementos en los distintos conjuntos de test selec
#
# set.seed(1)
# train.1 <- sample(c(TRUE, FALSE), size = nrow(cheddar), replace = TRUE, prob = c(0.7, 0.3))
# test.1 <- (!train.1)
# sum(test.1)
# model.all1 <- lm(taste ~ ., data = cheddar[train.1,])
#
# set.seed(1100)
# train.2 <- sample(c(TRUE, FALSE), size = nrow(cheddar), replace = TRUE, prob = c(0.7, 0.3))
# test.2 <- (!train.2)
# sum(test.2)
# sum(test.1==TRUE & test.2==TRUE)#solo 1 preseguimos
# model.all2 <- lm(taste ~ ., data = cheddar[train.2,])
#
# set.seed(5)
# train.3 <- sample(c(TRUE, FALSE), size = nrow(cheddar), replace = TRUE, prob = c(0.7, 0.3))
# test.3 <- (!train.3)
# sum(test.3)
# sum(test.1==TRUE & test.3==TRUE)
# sum(test.2==TRUE & test.3==TRUE)
```

```
# model.all3 <- lm(taste ~ ., data = cheddar[train.3,])
#
#
#
# # i) BACKWARD (alpha=0.05)
#
# drop1(model.all1, test = "F")
# # quitamos Acetic del modelo por ser la de mayor p-valor
#
# model.updateB1 <- update(model.all1, . ~ . - Acetic)
# drop1(model.updateB1, test = "F")
# # quitamos H2S del modelo por ser la de mayor p-valor
#
# model.updateB2 <- update(model.updateB1, . ~ . - H2S)
# drop1(model.updateB2, test = "F")
# # termina el proceso pues el p-valor de Lactic no llega a alpha
#
# model.1 <- lm(taste ~ Lactic, data = cheddar[train.1,])
# summary(model.1)
#
# # Para evitar repetir este proceso usamos el package mixlm con el que obtenemos los mismos resultados
# model.1 <- mixlm::backward(model.all1, alpha=0.05)
# summary(model.1) # LACTIC
#
# model.2 <- mixlm::backward(model.all2, alpha=0.05)
# summary(model.2) # H2S + LACTIC
#
# model.3 <- mixlm::backward(model.all3, alpha=0.05)
# summary(model.3) # H2S + LACTIC
#
#
#
# # ii) FORWARD (alpha=0.05)
# SCOPE <- (~ . + Acetic + H2S + Lactic)
# model.inicial <- lm(taste ~ 1, data = cheddar[train.1,]) # solo con termino independiente
#
# add1(model.inicial, scope = SCOPE, test = "F")
# # Añadimos Lactic por ser la variable predictora con menor p-valor
# model.updateF1 <- update(model.inicial, . ~ . + Lactic)
#
# add1(model.updateF1, scope = SCOPE, test = "F")
# # No añadimos ninguna variable pues todos los p-valores superan la barrera de alpha
#
# model.1a <- lm(taste ~ Lactic, data = cheddar[train.1,])
# summary(model.1a)
#
#
# # Al igual que en BACKWARD usamos el paquete mixlm para automatizar el proceso de FORWARD.
# model.1a <- mixlm::forward(model.all1, alpha=0.05)
# summary(model.1a) # LACTIC
#
# model.2a <- mixlm::forward(model.all2, alpha=0.05)
# summary(model.2a) # H2S + LACTIC
```

```
#
# model.3a <- mixlm::forward(model.all3, alpha=0.05)
# summary(model.3a) # H2S + LACTIC
#
# # Nótese que los modelos obtenidos por BACKWARD y FORWARD coinciden para cada conjunto train.
#
#
#
# # iii) CRITERIOS
#
# # R2 ajustado
# models1 <- regsubsets(taste ~ ., data = cheddar[train.1,])
# models2 <- regsubsets(taste ~ ., data = cheddar[train.2,])
# models3 <- regsubsets(taste ~ ., data = cheddar[train.3,])
#
# summary(models1)
# summary(models2)
# summary(models3)
#
# MR2adj1 <- summary(models1)$adjr2
# MR2adj2 <- summary(models2)$adjr2
# MR2adj3 <- summary(models3)$adjr2
#
# summary(models1)$which[which.max(MR2adj1), ]
# summary(models2)$which[which.max(MR2adj2), ]
# summary(models3)$which[which.max(MR2adj3), ]
#
#
#
# # Cp de Mallows
# MCp1 <- summary(models1)$cp
# MCp2 <- summary(models2)$cp
# MCp3 <- summary(models3)$cp
#
#
# summary(models1)$which[which.min(MCp1), ]
# summary(models2)$which[which.min(MCp2), ]
# summary(models3)$which[which.min(MCp3), ]
#
#
# # Criterio de Informacion de Bayes (BIC)
# MBIC1 <- summary(models1)$bic
# MBIC2 <- summary(models2)$bic
# MBIC3 <- summary(models3)$bic
#
# summary(models1)$which[which.min(MBIC1), ]
# summary(models2)$which[which.min(MBIC2), ]
# summary(models3)$which[which.min(MBIC3), ]
#
# # Criterio de Informacion de Akaike (AIC)
# stepAIC(model.all1, scope = SCOPE, k = 2)
# stepAIC(model.all2, scope = SCOPE, k = 2)
# stepAIC(model.all3, scope = SCOPE, k = 2)
```

```
#
#
# # Para el conjunto train.1 aparecen modelos con Lactic y H2S + Lactic como variables predictoras,
# #  mientras que para los conjuntos train.2 y train.3 unicamente aparece el modelo taste ~ H2S + Lacti
# model.1crit <- lm(taste ~ H2S + Lactic, data=cheddar[train.1,])
#
# # Esto nos puede llevar a pensar que el modelo final vaya a ser el que utiliza H2S y Lactic.
#
# model.1
# model.1crit
# model.2
# model.3
#
# anova(model.1,model.all1)
# anova(model.1crit,model.all1)
# anova(model.2,model.all2)
# anova(model.3,model.all3)
# # De esta forma vemos que los modelos creados son mejores que los completos.
#
#
#
#
#
# # 4) Diagnostico. Comprobaciones de hipotesis, outliers y observaciones influyentes
#
# # Comprobacion de hipotesis de los  modelos
#
# # Linealidad
#
# resettest(model.1, power=2:3, type="regressor", data=cheddar[train.1,])
# resettest(model.1crit, power=2:3, type="regressor", data=cheddar[train.1,])
# resettest(model.2, power=2:3, type="regressor", data=cheddar[train.2,])
# resettest(model.3, power=2:3, type="regressor", data=cheddar[train.3,])
# # p-valores > 0.05 luego aceptamos hipótesis de linealidad en todos los casos
#
# plot(cheddar$H2S, cheddar$taste,
#      main = "Relacion entre Taste y H2S",
#      xlab = "H2S", ylab = "Taste",
#      pch = 19, frame = FALSE)
# plot(cheddar$Lactic, cheddar$taste,
#      main = "Relacion entre Taste y Lactic",
#      xlab = "Lactic", ylab = "Taste",
#      pch = 19, frame = FALSE)
#
#
#
# # Normalidad y Autocorrelacion
#
# shapiro.test(resid(model.1)) # p-valor > 0.05 no se rechaza la hipótesis nula, i.e. la normalidad
# qqnorm(resid(model.1))
# qqline(resid(model.1))
#
# shapiro.test(resid(model.1crit))#bien
```

```
# qqnorm(resid(model.1crit))
# qqline(resid(model.1crit))
#
# shapiro.test(resid(model.2))#bien
# qqnorm(resid(model.2))
# qqline(resid(model.2))
#
# shapiro.test(resid(model.3))#bien
# qqnorm(resid(model.3))
# qqline(resid(model.3))
#
# # Observamos que las colas no siguen el mismo patrón que el resto de datos, lo que podría indicar que
# #  el modelo no sigue una distribución normal. Sin embargo, al no afectar a un gran porcentaje de los
# #  teniendo en cuenta el resultado previo obtenido por el Test de Shapiro-Wilk no rechazamos la hipót
#
#
# # Media de errores nula
#
# t.test(resid(model.1), mu = 0, alternative = "two.sided")
# t.test(resid(model.1crit), mu = 0, alternative = "two.sided")
# t.test(resid(model.2), mu = 0, alternative = "two.sided")
# t.test(resid(model.3), mu = 0, alternative = "two.sided")
# # p-valores ~ 1 > 0.05 luego aceptamos la hipotesis nula en todos los casos
#
#
#
# # Autocorrelacion
#
# durbinWatsonTest(model.1)
# durbinWatsonTest(model.1crit)
# durbinWatsonTest(model.2)
# durbinWatsonTest(model.3)
# # p-valores > 0.05 luego rechazamos que haya autocorrelacion
#
#
#
# # Homocedasticidad
#
# fmodel1 <- fortify(model.1)
# fmodel1crit <- fortify(model.1crit)
# fmodel2 <- fortify(model.2)
# fmodel3 <- fortify(model.3)
#
# X1 <- fmodel1$.fitted
# Y1 <- fmodel1$.stdresid
#
# X1crit <- fmodel1crit$.fitted
# Y1crit <- fmodel1crit$.stdresid
#
# X2 <- fmodel2$.fitted
# Y2 <- fmodel2$.stdresid
#
# X3 <- fmodel3$.fitted
```

```
# Y3 <- fmodel3$.stdresid
#
# plot(X1, Y1, ylab = "Residuos estandarizados", xlab = "valores ajustados")
# plot(X1crit, Y1crit, ylab = "Residuos estandarizados", xlab = "valores ajustados")
# plot(X2, Y2, ylab = "Residuos estandarizados", xlab = "valores ajustados")#casi casi hay patron
# plot(X3, Y3, ylab = "Residuos estandarizados", xlab = "valores ajustados")
# # Los residuos se distribuyen de forma homogénea a lo largo de una banda horizontal, luego se verific
#
#
# ncvTest(model.1)
# ncvTest(model.1crit)
# ncvTest(model.2)
# ncvTest(model.3)
# # p-valores > 0.05, luego no hay evidencia para rechazar que la varianza sea constante
#
#
#
# # Outliers
#
# alpha <- 0.05
# n1 <- nrow(cheddar[train.1,])
# p1 <- nrow(summary(model.1)$coef)
#
# n1crit <- nrow(cheddar[train.1,])
# p1crit <- nrow(summary(model.1crit)$coef)
#
# n2 <- nrow(cheddar[train.2,])
# p2 <- nrow(summary(model.2)$coef)
#
# n3 <- nrow(cheddar[train.3,])
# p3 <- nrow(summary(model.3)$coef)
#
#
# # El valor critico de Bonferroni t_{1-alpha/2n;n-p-1}
# BCV1 <- qt(1 - alpha / (2 * n1), n1-p1-1)
# BCV1crit <- qt(1 - alpha / (2 * n1crit), n1crit-p1crit-1)
# BCV2 <- qt(1 - alpha / (2 * n2), n2-p2-1)
# BCV3 <- qt(1 - alpha / (2 * n3), n3-p3-1)
#
# sum(abs(rstudent(model.1)) > BCV1)
# sum(abs(rstudent(model.1crit)) > BCV1crit)
# sum(abs(rstudent(model.2)) > BCV2)
# sum(abs(rstudent(model.3)) > BCV3)
#
# outlierTest(model.1)#NA?
# outlierTest(model.1crit) #NA?
# outlierTest(model.2)
# outlierTest(model.3)
# # No hay outliers en ninguno de los dos modelos planteados
#
#
#
# # Observaciones Influyentes
```

```
#
# # Criterio 1: valores leverage (hii) mayores que 2p/n
#
# X1 <- model.matrix(~ H2S+Lactic, data = cheddar[train.1,])
# X1crit <- model.matrix(~ H2S+Lactic, data = cheddar[train.1,])
# X2 <- model.matrix(~ H2S+Lactic, data = cheddar[train.2,])
# X3 <- model.matrix(~ Lactic, data = cheddar[train.3,])
#
# H1 <- X1 %*% solve(t(X1) %*% X1) %*% t(X1)
# H1crit <- X1crit %*% solve(t(X1crit) %*% X1crit) %*% t(X1crit)
# H2 <- X2 %*% solve(t(X2) %*% X2) %*% t(X2)
# H3 <- X3 %*% solve(t(X3) %*% X3) %*% t(X3)
#
# hii1 <- diag(H1)
# hii1crit <- diag(H1crit)
# hii2 <- diag(H2)
# hii3 <- diag(H3)
#
# hCV1 <- 2 * p1 / n1
# hCV1crit <- 2 * p1crit / n1crit
# hCV2 <- 2 * p2 / n2
# hCV3 <- 2 * p3 / n3
#
# sum(hii1 > hCV1)
# which(hii1>hCV1)#1  5 16 23 24 26
#
# sum(hii1crit > hCV1crit)
# which(hii1crit>hCV1crit)
#
# sum(hii2 > hCV2)
# which(hii2>hCV2)
#
# sum(hii3 > hCV3)
#
#
# # Criterio 2: valores |DFFITS| son mayores que 2*sqrt(p/n)
#
# dffitsCV1 <- 2 * sqrt(p1 / n1)
# dffitsCV1crit <- 2 * sqrt(p1crit / n1crit)
# dffitsCV2 <- 2 * sqrt(p2 / n2)
# dffitsCV3 <- 2 * sqrt(p3 / n3)
#
# dffitsmodel1 <- dffits(model.1)
# dffitsmodel1crit <- dffits(model.1crit)
# dffitsmodel2 <- dffits(model.2)
# dffitsmodel3 <- dffits(model.3)
#
# sum(dffitsmodel1 > dffitsCV1)
# which(dffitsmodel1 > dffitsCV1)#1 12 24
#
# sum(dffitsmodel1crit > dffitsCV1crit)
# which(dffitsmodel1crit > dffitsCV1crit)#1, 12
#
```

```
# sum(dffitsmodel2 > dffitsCV2)
# sum(dffitsmodel3 > dffitsCV3)
#
#
#
# # Criterio 3: valores |DFBETAS| mayores que 2/sqrt(n)
#
# dfbetaCV1 <- 2 / sqrt(n1)
# dfbetaCV1crit <- 2 / sqrt(n1crit)
# dfbetaCV2 <- 2 / sqrt(n2)
# dfbetaCV3 <- 2 / sqrt(n3)
#
# dfbetamodel1 <- dfbeta(model.1)
# dfbetamodel1crit <- dfbeta(model.1crit)
# dfbetamodel2 <- dfbeta(model.2)
# dfbetamodel3 <- dfbeta(model.3)
#
#
# sum(dfbetamodel1[, 1] > dfbetaCV1)
# sum(dfbetamodel1[, 2] > dfbetaCV1)
# which(dfbetamodel1[, 1] > dfbetaCV1)
# which(dfbetamodel1[, 2] > dfbetaCV1)
#
# sum(dfbetamodel1crit[, 1] > dfbetaCV1crit)
# sum(dfbetamodel1crit[, 2] > dfbetaCV1crit)
# sum(dfbetamodel1crit[, 3] > dfbetaCV1crit)
# which(dfbetamodel1crit[, 1] > dfbetaCV1crit)
# which(dfbetamodel1crit[, 3] > dfbetaCV1crit)
#
# sum(dfbetamodel2[, 1] > dfbetaCV2)
# sum(dfbetamodel2[, 2] > dfbetaCV2)
# sum(dfbetamodel2[, 3] > dfbetaCV2)
# which(dfbetamodel2[, 1] > dfbetaCV2)
# which(dfbetamodel2[, 3] > dfbetaCV2)
#
# sum(dfbetamodel3[, 1] > dfbetaCV3)
# sum(dfbetamodel3[, 2] > dfbetaCV3)
# sum(dfbetamodel3[, 3] > dfbetaCV3)
# which(dfbetamodel3[, 1] > dfbetaCV3)
# which(dfbetamodel3[, 3] > dfbetaCV3)
#
# # Dado que nuestro datset es pequeño, no tendremso en cuenta este metodo por dar demasiadas observaci
#
#
# # Grafica con las distancias de Cook
# influencePlot(model.1)
# pos_influyentes_1 <- c(1,12,24)
#
# influencePlot(model.1crit)
# pos_influyentes_1crit <- c(1,8,12,23)
#
# influencePlot(model.2)
# pos_influyentes_2 <- c(1,6,7,8,15)
```

```
#
# influencePlot(model.3)
# pos_influyentes_3 <- c(1,7,15,19,24)
#
#
#
# # Colinealidad
# vif(model.1) # tiene sentido pues en model.1 solo hay una variable predictora
# vif(model.1crit)
# vif(model.2)
# vif(model.3) # como los valores de VIF no son grando, no indican colinealidad grave
#
#
# # Vamos a tomar como posibles influyentes las que aparecen por varios metodos en cada modelo
#
# # En el 1: 1 12
# pos_1 <- c(TRUE,rep(FALSE,10),TRUE,rep(FALSE,18))
# train.1inf <- train.1&!pos_1
# model.1inf <- lm(taste ~ Lactic, data = cheddar[train.1inf,])
#
# ncvTest(model.1)
# ncvTest(model.1inf) # empeora pero sigue cumpliendo la hipotesis
#
# shapiro.test(resid(model.1))
# shapiro.test(resid(model.1inf)) # mejora poco
#
# durbinWatsonTest(model.1)
# durbinWatsonTest(model.1inf) # mejora
#
# resettest(model.1, power=2:3, type="regressor", data=cheddar[train.1inf,])
# resettest(model.1inf, power=2:3, type="regressor", data=cheddar[train.1inf,]) # mejora bastante
#
# # Redefinimos el modelo eliminando las observaciones influyentes de pos_1.
# train.1aux<-train.1
# train.1 <- train.1inf
# model.1 <- model.1inf
#
#
# # En el 1crit: 1 12  23
# pos_1crit <- c(TRUE,rep(FALSE,10),TRUE,rep(FALSE,10),TRUE,rep(FALSE,7))
# train.1critinf <- train.1&!pos_1crit
# model.1critinf <- lm(taste ~H2S+Lactic , data = cheddar[train.1critinf,])
#
# ncvTest(model.1crit)
# ncvTest(model.1critinf) # empeora
#
# shapiro.test(resid(model.1crit))
# shapiro.test(resid(model.1critinf)) # mejora
#
# durbinWatsonTest(model.1crit)
# durbinWatsonTest(model.1critinf) # mejora bastante
#
# resettest(model.1crit, power=2:3, type="regressor", data=cheddar[train.1,])
```

```
# resettest(model.1critinf, power=2:3, type="regressor", data=cheddar[train.1critinf,]) # empeora dentr
#
# # Redefinimos el modelo eliminando las observaciones influyentes de pos_1crit.
# train.1crit <- train.1critinf
# model.1crit <- model.1critinf
#
#
# # En el 2: 6
# pos_2 <- c(rep(FALSE,5),TRUE,rep(FALSE,24))
# train.2inf <- train.2&!pos_2
# model.2inf <- lm(taste ~ H2S+Lactic, data = cheddar[train.2inf,])
#
# ncvTest(model.2)
# ncvTest(model.2inf) # mejora poco
#
# shapiro.test(resid(model.2))
# shapiro.test(resid(model.2inf)) # aproximadamente lo mismo
#
# durbinWatsonTest(model.2)
# durbinWatsonTest(model.2inf) # mejora
#
# resettest(model.2, power=2:3, type="regressor", data=cheddar[train.2inf,])
# resettest(model.2inf, power=2:3, type="regressor", data=cheddar[train.2inf,]) # aproximadamente lo mi
#
# # Redefinimos el modelo eliminando las observaciones influyentes de pos_2.
# train.2 <- train.2inf
# model.2 <- model.2inf
#
#
# # En el modelo 3 no hacemos comprobaciones nada ya que no se repiten observaciones por distintos meto
#
#
#
#
#
# # 5) Errores de Test. Comparacion de Modelos
#
# # Vamos a considerar otras dos semillas para evaluar los modelos y elegir el mejor.
# # Las tomamos de forma que no coincidan en la medida de lo posible los tests de las distintas partici
# # Dividimos el conjunto total para las dos semillas nuevas.
#
# set.seed((2234))
# train.4 <- sample(c(TRUE, FALSE), size = nrow(cheddar), replace = TRUE, prob = c(0.7, 0.3))
# test.4 <- (!train.4)
# sum(test.4)
# sum(test.1==TRUE & test.4==TRUE)
# sum(test.2==TRUE & test.4==TRUE)
# sum(test.3==TRUE & test.4==TRUE)
#
#
# set.seed((131))
# train.5 <- sample(c(TRUE, FALSE), size = nrow(cheddar), replace = TRUE, prob = c(0.7, 0.3))
# test.5 <- (!train.5)
```

```
# sum(test.5)
# sum(test.1==TRUE & test.5==TRUE)
# sum(test.2==TRUE & test.5==TRUE)
# sum(test.3==TRUE & test.5==TRUE)
# sum(test.4==TRUE & test.5==TRUE)
#
# train.1h <- train.1crit
# train.2h <- train.2
# train.3h <- train.3
# train.4h <- train.4
# train.5h <- train.5
#
# # Recalculamos train.2 porque fue modificado
# set.seed(1100)
# train.2 <- sample(c(TRUE, FALSE), size = nrow(cheddar), replace = TRUE, prob = c(0.7, 0.3))
#
#
#
#
# # Veamso si podemos eliminar posibles influyentes en los modelos restantes
#
# mh1 <- lm(taste ~ H2S + Lactic, data = cheddar[train.1h,]) # comprobados supuestos
# mh2 <- lm(taste ~ H2S + Lactic, data = cheddar[train.2h,]) # comprobados supuestos
# mh3 <- lm(taste ~ H2S + Lactic, data = cheddar[train.3h,]) # comprobados supuestos
# mh4 <- lm(taste ~ H2S + Lactic, data = cheddar[train.4h,])
# mh5 <- lm(taste ~ H2S + Lactic, data = cheddar[train.5h,])
#
# m1 <- lm(taste ~ Lactic, data = cheddar[train.1,]) # comprobados supuestos
# m2 <- lm(taste ~ Lactic, data = cheddar[train.2,])
# m3 <- lm(taste ~ Lactic, data = cheddar[train.3,])
# m4 <- lm(taste ~ Lactic, data = cheddar[train.4,])
# m5 <- lm(taste ~ Lactic, data = cheddar[train.5,])
#
#
# # Al igual que hicimos en 4 comprobamos si retirando outliers y observaciones influyentes mejoran las
# #  Unicamente consideraremos observaciones obtenidas tanto por outlierTest() como por influencePlot()
#
# # mh4
# outlierTest(mh4)
# influencePlot(mh4) # 6,12,15,24
# pos4h <- c(rep(FALSE,5),TRUE,rep(FALSE,5),TRUE,rep(FALSE,2),TRUE,rep(FALSE,8),TRUE,rep(FALSE,6))
#
# train.4inf <- train.4h&!pos4h
# mh4inf <- lm(taste ~ H2S+Lactic, data = cheddar[train.4inf,])
#
# ncvTest(mh4)
# ncvTest(mh4inf) # mejora MUCHO
#
# shapiro.test(resid(mh4))
# shapiro.test(resid(mh4inf)) # mejora
#
# durbinWatsonTest(mh4)
# durbinWatsonTest(mh4inf) # mejora
```

```
#
# resettest(mh4, power=2:3, type="regressor", data=cheddar[train.4inf,])
# resettest(mh4inf, power=2:3, type="regressor", data=cheddar[train.4inf,]) # empeora pero es aceptable
#
# # Redefinimos eliminando pos4h por ser tanta la mejora en homocedasticidad que compensa la linealidad
# train.4h <- train.4inf
# mh4 <- mh4inf
#
#
# # mh5
# outlierTest(mh5)
# influencePlot(mh5) # 6,7,8,12,15
# pos5h <- c(rep(FALSE,5),TRUE,TRUE,TRUE,rep(FALSE,3),TRUE,rep(FALSE,2),TRUE,rep(FALSE,15))
#
# train.5inf <- train.5h&!pos5h
# mh5inf <- lm(taste ~ H2S+Lactic, data = cheddar[train.5inf,])
#
# ncvTest(mh5)
# ncvTest(mh5inf) # mejora
#
# shapiro.test(resid(mh5))
# shapiro.test(resid(mh5inf)) # empeora
#
# durbinWatsonTest(mh5)
# durbinWatsonTest(mh5inf) # a veces no se cumple la hipotesis nula de no autocorrelacion
#
# resettest(mh5, power=2:3, type="regressor", data=cheddar[train.5inf,])
# resettest(mh5inf, power=2:3, type="regressor", data=cheddar[train.5inf,]) # empeora
#
# # Decidimos no actualizar el modelo pues no siempre se rechaza la correlacion y empeoran otros p-valo
#
#
# # m2
# outlierTest(m2)
# influencePlot(m2) # 1,12,15,18,24
# pos2 <- c(TRUE,rep(FALSE,10),TRUE,rep(FALSE,2),TRUE,rep(FALSE,2),TRUE,rep(FALSE,5),TRUE,rep(FALSE,6))
#
# train.2inf <- train.2&!pos2
# m2inf <- lm(taste ~ Lactic, data = cheddar[train.2inf,])
#
# ncvTest(m2)
# ncvTest(m2inf) # empeora ligeramente
#
# shapiro.test(resid(m2))
# shapiro.test(resid(m2inf)) # empeora ligeramente
#
# durbinWatsonTest(m2)
# durbinWatsonTest(m2inf) # empeora bastante
#
# resettest(m2, power=2:3, type="regressor", data=cheddar[train.2inf,])
# resettest(m2inf, power=2:3, type="regressor", data=cheddar[train.2inf,]) # mejora
#
# # Decicimos no redefinirlo, ya que empeora en muchas y mejora bastante en una
```

```
#
#
# # m3
# outlierTest(m3)
# influencePlot(m3) # 1,15,19,24
# pos3 <- c(TRUE,rep(FALSE,13),TRUE,rep(FALSE,3),TRUE,rep(FALSE,4),TRUE,rep(FALSE,6))
#
# train.3inf <- train.3[!pos3
# m3inf <- lm(taste ~ Lactic, data = cheddar[train.3inf,])
#
# ncvTest(m3)
# ncvTest(m3inf) # mejora poco
#
# shapiro.test(resid(m3))
# shapiro.test(resid(m3inf)) # mejora
#
# durbinWatsonTest(m3)
# durbinWatsonTest(m3inf) # empeora
#
# resettest(m3, power=2:3, type="regressor", data=cheddar[train.3inf,])
# resettest(m3inf, power=2:3, type="regressor", data=cheddar[train.3inf,]) # empeora casi lo pierde
#
# # Decidimos no redefinirlo por estar al borde en un criterio y no mejorar mucho en otros
#
#
# # m4
# outlierTest(m4)
# influencePlot(m4) # 6,12,15,20,24
# pos4 <- c(rep(FALSE,5),TRUE,rep(FALSE,5),TRUE,rep(FALSE,2),TRUE,rep(FALSE,4),TRUE,rep(FALSE,3),TRUE,r
#
# train.4inf <- train.4[!pos4
# m4inf <- lm(taste ~ Lactic, data = cheddar[train.4inf,])
#
# ncvTest(m4)
# ncvTest(m4inf) # empeora bastante
#
# shapiro.test(resid(m4))
# shapiro.test(resid(m4inf)) # empeora
#
# durbinWatsonTest(m4)
# durbinWatsonTest(m4inf)# empeora bastante
#
# resettest(m4, power=2:3, type="regressor", data=cheddar[train.4inf,])
# resettest(m4inf, power=2:3, type="regressor", data=cheddar[train.4inf,]) # mejora algo
#
# # Decidimos no cambiarlo porque empeora en casi todo, y de por si tenia buenos p-values
#
#
# # m5
# outlierTest(m5)
# influencePlot(m5) # 1,8,12,15,18,24
# pos5 <- c(TRUE,rep(FALSE,6),TRUE,rep(FALSE,3),TRUE,rep(FALSE,2),TRUE,rep(FALSE,2),TRUE,rep(FALSE,5),T
#
```

```
# train.5inf <- train.5&!pos5
# m5inf <- lm(taste ~ Lactic, data = cheddar[train.5inf,])
#
# ncvTest(m5)
# ncvTest(m5inf) # empeora bastante
#
# shapiro.test(resid(m5))
# shapiro.test(resid(m5inf)) # mejora
#
# durbinWatsonTest(m5)
# durbinWatsonTest(m5inf) # no se verifica la hipotesis nula de no autocorrelacion
#
# resettest(m5, power=2:3, type="regressor", data=cheddar[train.5inf,])
# resettest(m5inf, power=2:3, type="regressor", data=cheddar[train.5inf,]) # empeora
#
# # Decidimos no cambiarlo por perderse la correlación
#
# # Notese que en los originales siempre se cumplian las hipotesis, aunque no era necesario para el pro
#
#
#
#
# lista_test <- list(test.1,test.2,test.3,test.4,test.5)
# lista_trainh <- list(train.1h,train.2h,train.3h,train.4h,train.5h)
# lista_train <- list(train.1,train.2,train.3,train.4,train.5)
#
#
# modelos_hip <- c("S1 taste ~ H2s + Lactic",
#                  "S1 taste ~ Lactic",
#                  "S2 taste ~ H2s + Lactic",
#                  "S2 taste ~ Lactic",
#                  "S3 taste ~ H2s + Lactic",
#                  "S3 taste ~ Lactic",
#                  "S4 taste ~ H2s + Lactic",
#                  "S4 taste ~ Lactic",
#                  "S5 taste ~ H2s + Lactic",
#                  "S5 taste ~ Lactic",
#                  "Nivel de significacion")
#
# hip_RL <- c("Distribución_normal",
#             "Media_0",
#             "Varianza_no_constante",
#             "No_Autocorrelación")
# placeholder <- vector(mode = "logical",length = 11)
# df_hipRL <- data.frame("0" = placeholder,
#                        "1" = placeholder,
#                        "2" = placeholder,
#                        "3" = placeholder,
#                        row.names = modelos_hip)
#
# colnames(df_hipRL) <- hip_RL
#
# r <- 1
```

```
# for (dtrain in lista_trainh){
#   model.HL.lm <- lm(taste ~ H2S + Lactic,
#                     data = cheddar[dtrain,])
#   residuos <- resid(model.HL.lm)
#   new_row <- c()
#
#   shap <- round(shapiro.test(residuos)$p.value,4)
#   t <- t.test(residuos, mu = 0, alternative = "two.sided")$p.value
#   v <- round(ncvTest(model.HL.lm)$p,4)
#   dw <- round(durbinWatsonTest(model.HL.lm)$p,4)
#
#   new_row = c(shap,t,v,dw)
#   df_hipRL[r,] <- new_row
#   r = r + 2
# }
#
#
# r <- 2
# for (dtrain in lista_train){
#   model.L.lm <- lm(taste ~Lactic,
#                    data = cheddar[dtrain,])
#   residuos <- resid(model.L.lm)
#   new_row <- c()
#
#   shap <- round(shapiro.test(residuos)$p.value,4)
#   t <- t.test(residuos, mu = 0, alternative = "two.sided")$p.value
#   v <- round(ncvTest(model.L.lm)$p,4)
#   dw <- round(durbinWatsonTest(model.L.lm)$p,4)
#
#   new_row = c(shap,t,v,dw)
#   df_hipRL[r,] <- new_row
#   r = r + 2
# }
# df_hipRL[11,] <- c(rep(0.05,4))
# df_hipRL # Todos los modelos cumplen las hipótesis
#
#
#
# err1 <- 0
# for (i in 5){
#   dtrain <- lista_trainh[[i]]
#   dtest <- lista_test[[i]]
#   dmod <- lm(taste ~ H2S + Lactic, data = cheddar[dtrain,])
#   Y <- cheddar[dtest,]$taste
#   Yhat <- predict(obj = dmod, newdata = cheddar[dtest,])
#   err1 <- err1 + mean((Y - Yhat)^2)
# }
# err1 <- err1/5
# err1 # H2S + LACTIC tiene error medio 11.63
#
#
# err2<-0
# for (i in 5){
```

```
#   dtrain <- lista_train[[i]]
#   dtest <- lista_test[[i]]
#   dmod <- lm(taste ~ Lactic, data = cheddar[dtrain,])
#   Y <- cheddar[dtest,]$taste
#   Yhat <- predict(obj = dmod, newdata = cheddar[dtest,])
#   err2 <- err2 + mean((Y - Yhat)^2)
# }
# err2 <- err2/5
# err2 # LACTIC tiene error medio 19.4
#
# err1 < err2 # Concluimos que el modelo taste ~ H2S + Lactic es mejor que taste ~ Lactic.
#
#
#
#
#
# # 6) Conclusion
#
# model.y <- lm(taste ~ H2S + Lactic,data=cheddar) # tomamos el mejor modelo de los obtenidos en 4
# summary(model.y)
# plot(model.y)
#
# # Veamos los posibles outliers
# outlierTest(model.y)
# influencePlot(model.y)
#
#
# # Estudio de hipótesis supuestas
#
# shapiro.test(resid(model.y)) # normalidad de residuos
# t.test(resid(model.y), mu = 0, alternative = "two.sided") # media cero de los residuos
# ncvTest(model.y) # varianza constante de los residuos
# durbinWatsonTest(model.y) # no autocorrelacion
# resettest(model.y ,power=2:3, type="regressor", data=cheddar) # linealidad
#
# # Efectivamente, cumple todas las hipótesis
#
#
# summary(model.y) # en el summary observamos los valores de betahat, sus p-valores y sigma
#
# coeff <- summary(model.y)$coeff[,1]
# coeff
#
# # Calculo de intervalo de confianza de beta1(H2S) y beta2(Lactic)
#
# # Metodo Bonferroni
# alpha <- 0.10
# summary(model.y)$coef
# b <- summary(model.y)$coef[2:3, 1]
# s.b <- summary(model.y)$coef[2:3, 2]
# g <- 3
# n <- nrow(cheddar)
# p <- ncol(summary(model.y)$coef)
```

```
# t_teo <- qt(1 - alpha / (2 * g), n - p)
# BomSimCI <- matrix(c(b - t_teo * s.b, b + t_teo * s.b), ncol = 2)
# conf <- c("5%", "95%")
# bnam <- c("H2S", "Lactic")
# dimnames(BomSimCI) <- list(bnam, conf)
# BomSimCI
#
# # Intervalo de confianza simultaneo por el metodo de Scheffe
# Q <- p - 1
# f_teo <- qf(0.9, Q, n - p)#0.9 no seria 0.95?
# SchSimCI <- matrix(c(b - sqrt(Q * f_teo) * s.b, b + sqrt(Q * f_teo) * s.b), ncol = 2)
# conf <- c("5%", "95%")
# bnam <- c("H2S", "Lactic")
# dimnames(SchSimCI) <- list(bnam, conf)
# SchSimCI
#
#
# # La ecuación de nuestro modelo final es y = (-27.6) + 3.95*x_H + 19.9*x_L,
# #    donde x_H y x_L denotan los valores observados de H2S y Lactic.
#
#
# rse <- sqrt(deviance(model.y)/df.residual(model.y))
# rse  # varianza de los residuos
#
# pval <- summary(model.y)$coeff[,4]
# pval  # vector con los p-valores
#
#
# # Calculo de su R^2
# anova(model.y)
# SSE <- anova(model.y)[3,2]
# SST <- anova(model.y)[1,2]+anova(model.y)[2,2]+anova(model.y)[3,2]
# rsqr <- 1 - SSE/SST
# rsqr
# summary(model.y)$r.squared# es la misma
#
# # Calculo de su R^2 ajustada
# MSE <- SSE/anova(model.y)[3,1]
# MST <- SST/(anova(model.y)[1,1]+anova(model.y)[2,1]+anova(model.y)[3,1])
# Radj <-1-MSE/MST
# Radj  # se comprueba en la tabla summary que el valor es correcto
#
#
# # Observamos como se distribuye la variable taste en función de H2S y Lactic
# plot_ly(x=H2S, y=Lactic, z=taste, type="scatter3d", mode="marker", color=taste) %>%
#   layout(scene = list(xaxis = list(title = 'H2S (%)'),
#                       yaxis = list(title = 'Lactic (%)'),
#                       zaxis = list(title = 'Taste (0-100)')))
#
#
# # Vemos el plano de regresion del modelo propuesto.
# # En rojo se marcan 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)')
# planereg$plane3d(model.y, lty.box="solid", col='mediumblue')
```