Índice

1. Trabajo sobre Análisis de Datos **2**
1.1. California . . . 2
1.1.1. Definición del problema . . . 2
1.1.2. Descripción de los datos . . . 2
1.1.3. Procesamiento de los datos . . . 3
1.1.4. Resumen de los datos . . . 4
1.1.5. Correlación y reducción de la dimensionalidad . . . 9
1.1.6. Transformación de los datos . . . 10
1.1.7. Conclusiones . . . 12
1.2. Bupa . . . 12
1.2.1. Definición del problema . . . 12
1.2.2. Descripción de los datos . . . 13
1.2.3. Procesamiento de los datos . . . 14
1.2.4. Resumen de los datos . . . 15
1.2.5. Correlación y reducción de dimensionalidad . . . 21
1.2.6. Transformación de los datos . . . 23
1.2.7. Conclusiones . . . 24

2. Trabajo sobre Regresión **25**
2.1. Regresión lineal simple . . . 25
2.2. Regresión múltiple . . . 26
2.3. k-NN para regresión . . . 27
2.4. Comparación de los algoritmos . . . 28

3. Trabajo sobre Clasificación **29**
3.1. k-NN . . . 29
3.2. LDA . . . 31
3.3. QDA . . . 34
3.4. Comparación de los algoritmos . . . 35

A. Código del Trabajo sobre Análisis de Datos **37**
A.1. *california* . . . 37
A.2. *bupa* . . . 41

B. Código del Trabajo sobre Regresión **46**

C. Código del Trabajo sobre Clasificación **56**

1. Trabajo sobre Análisis de Datos

1.1. California

1.1.1. Definición del problema

Para el dataset *california* se tiene que resolver un problema de regresión. Dicho dataset que se analizará proviene de un conjunto de datos de viviendas en California de 1990 que contiene información sobre diferentes características de las viviendas en varias localidades de este estado y cuenta con las variables **Longitude, Latitude, HousingMedianAge, TotalRooms, TotalBedrooms, Population, Households, MedianIncome** y **MedianHouseValue**.

El objetivo a cumplir para este caso es crear un modelo de regresión que pueda predecir el valor mediano de las viviendas a partir de otras características, por lo tanto la variable dependiente se trata de **MedianHouseValue**, las demás son independientes.

Teniendo en cuenta todo esto se pueden realizar una serie de hipótesis que tienen que ver principalmente con la relación entre las distintas variables:

1. El ingreso (MedianIncome) está relacionado directamente con el valor de las casas (MedianHouseValue).
2. Las variables de longitud y latitud están correlacionadas debido a las zonas de aglomeración de las viviendas.
3. El número de dormitorios (TotalBedrooms) está relacionado directamente con el número de habitaciones (TotalRooms).
4. El número de casas (Households) está relacionado directamente con la población (Population).
5. La distribución de los ingresos (MedianIncome) está sesgada.

Más adelante, con el uso de gráficos, podremos verificar si estas hipótesis son acertadas y conforme a ello se procesarán los datos de manera conveniente.

1.1.2. Descripción de los datos

El dataset presenta formato *csv* separado por comas, para leerlo como un *data frame* se puede usar la función *read.csv*. Este contiene **20640** instancias y **9** características que se han mencionado ya anteriormente. Los tipos de datos atómicos que se tienen son:

1. **Longitude**: Longitud en grados (^{o}) de la localidad, tipo *numeric* (real).
2. **Latitude**: Latitud en grados (^{o}) de la localidad, tipo *numeric* (real).
3. **HousingMedianAge**: Edad mediana de la casa, tipo *integer* (entero).
4. **TotalRooms**: Número total de habitaciones de la casa, tipo *integer* (entero).
5. **TotalBedrooms**: Número total de dormitorios, tipo *integer* (entero).
6. **Population**: Población de la localidad, tipo *integer* (entero).
7. **Households**: Número de casa, tipo *integer* (entero).

8. **MedianIncome**: Ingreso mediano (en unidades de decenas de miles de $), tipo *numeric* (real).

9. **MedianHouseValue**: Valor mediano de la casa, tipo *integer* (entero).

Como vemos no hay variables categóricas, todas son cuantitativas.

Por otro lado, podemos hacernos una idea del contenido del dataset realizando un rápido estudio estadístico de las distintas variables, que se presenta a continuación en la tabla 1.

Característica	Media	Mediana	Mínimo	Máximo	1er Cuartil	3er Cuartil	Desv. Est.
Longitude	-119.6	-118.5	-124.4	-114.3	-121.8	-118.0	2.0
Latitude	35.63	34.26	32.54	41.95	33.93	37.71	2.14
HousingMedianAge	28.64	29.00	1.00	52.00	18.00	37.00	12.60
TotalRooms	2636	2127	2	39320	2636	3148	2182
TotalBedrooms	537.9	435.0	1.0	6445.0	295.0	647.0	421.0
Population	1425	1166	3	35682	787	1725	1132
Households	499.5	409.0	1.0	6082.0	280.0	605.0	382.0
MedianIncome	3.8707	3.5348	0.4999	15.0001	2.5634	4.7432	1.9000
MedianHouseValue	206856	179700	14999	500001	119600	264725	115396

Tabla 1: Estudio estadístico de las características de *california*.

1.1.3. Procesamiento de los datos

El dataset no presenta valores perdidos en ninguna de sus características, por lo tanto en ese sentido no es necesario realizar ningún tipo de imputación ni eliminación de variables o instancias.

En cuanto a los *outliers*, podemos identificarlos utilizando un gráfico de tipo *box plot* para cada variable. Este se muestra a continuación en la figura 1.

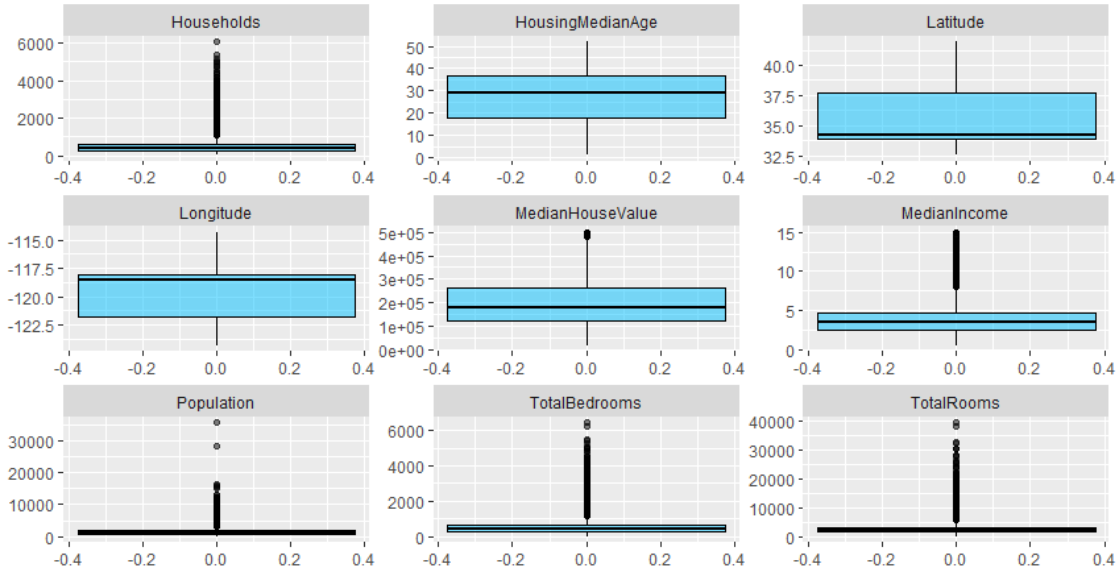

Figura 1: Distribución de datos para cada variable usando *box plots*.

Como se puede observar en la figura anterior, la mayoría de variables presentan *outliers*, las únicas que no presentan son Longitude, Latitude y HousingMedianAge. Sin embargo se ha optado por conservarlos debido a que se tratan de *outliers* multivariables cuyas causas son de tipo natural, por lo tanto son necesarios para que el modelo pueda captar adecuadamente toda la variabilidad que presenta el mercado.

1.1.4. Resumen de los datos

En esta sección se muestran gráficos univariables y bivariables para tener una perspectiva global del conjunto de datos con el que se está trabajando y ver como se relacionan las distintas características, así como otros gráficos más concretos que servirán para verificar las distintas hipótesis que se han hecho al principio de esta parte.

En la figura 2 encontramos los *plots* univariables, mientras que en la figura 3 se muestran los gráficos bivariables.

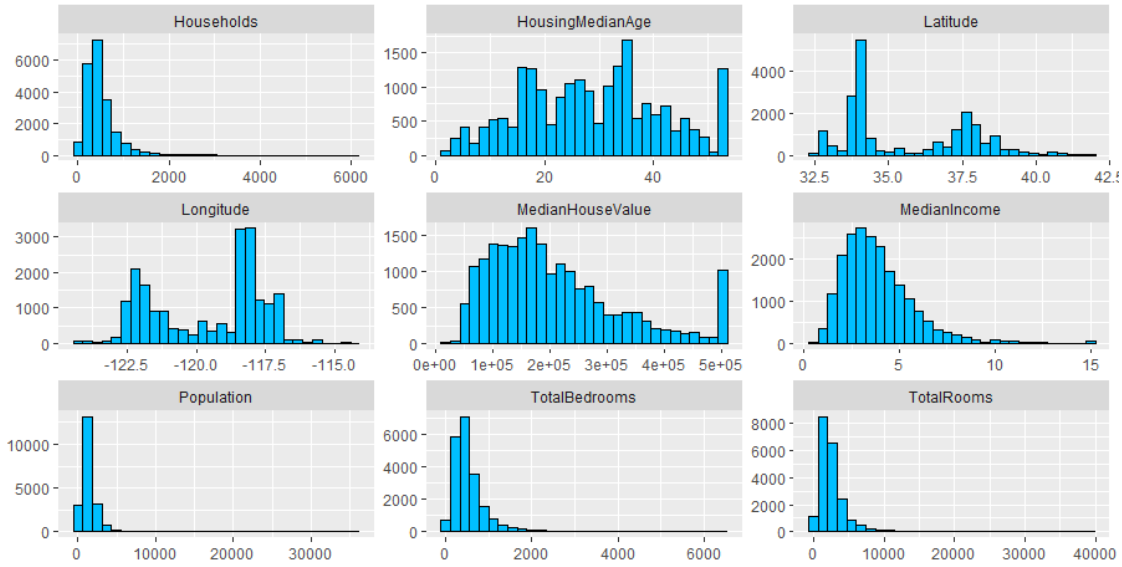

Figura 2: Histogramas de todas los atributos por separado.

Por otro lado, los plots bivariables se muestran en la figura 3.

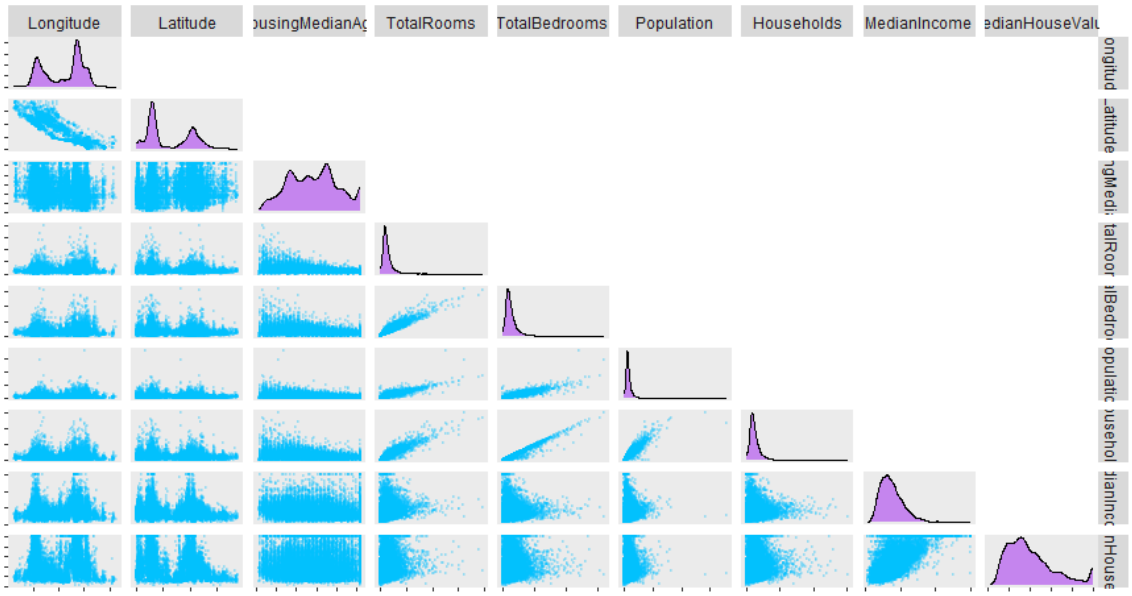

Figura 3: *Scatter plots* bivariables de todos los atributos.

En la figura 3 se ha optado por eliminar las escalas de los ejes, ya que el propósito principal de este gráfico es tener una perspectiva general sobre cómo se relacionan las características en parejas. Como se puede apreciar, esta relación entre algunas variables es más que evidente, pero eso es un aspecto que trataremos en el siguiente apartado, donde estudiaremos la correlación más detenidamente. En lo que respecta a la figura 2, podemos ver que tipo sesgo y curtosis tienen las variables. Algunas variables como **HouseHolds**, **TotalBedrooms** y **TotalRooms** tienen una alta curtosis y están sesgadas hacia la derecha. Por otro lado, características como **MedianHouseValue** y **MedianIncome** parecen tener una curtosis más moderada pero igualmente están sesgadas positivamente. Por último **Longitude** y **Latitude** parecen ser variables que siguen una distribución semejante a la bimodal.

En otro orden de cosas, a continuación se muestran gráficos relativos a las hipótesis que se han hecho cuando hemos definido el problema.

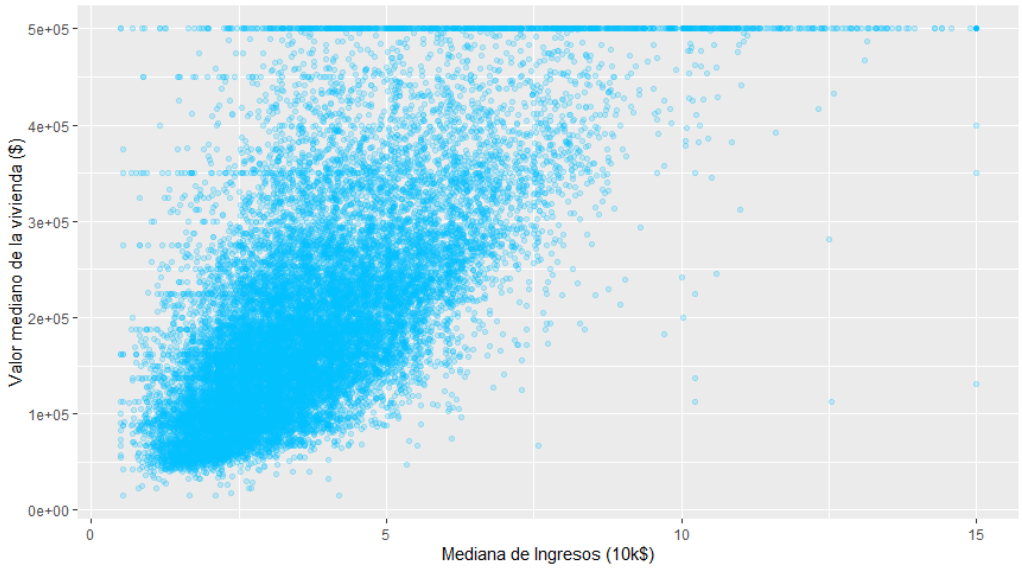

Figura 4: Gráfico que muestra los ingresos frente al valor de las viviendas.

La primera hipótesis que se ha realizado es que existe relación entre los ingresos y el valor de las viviendas. Como se puede observar en la figura 4, esta hipótesis se confirma, la relación que existe entre la variable dependiente y los ingresos es directa.

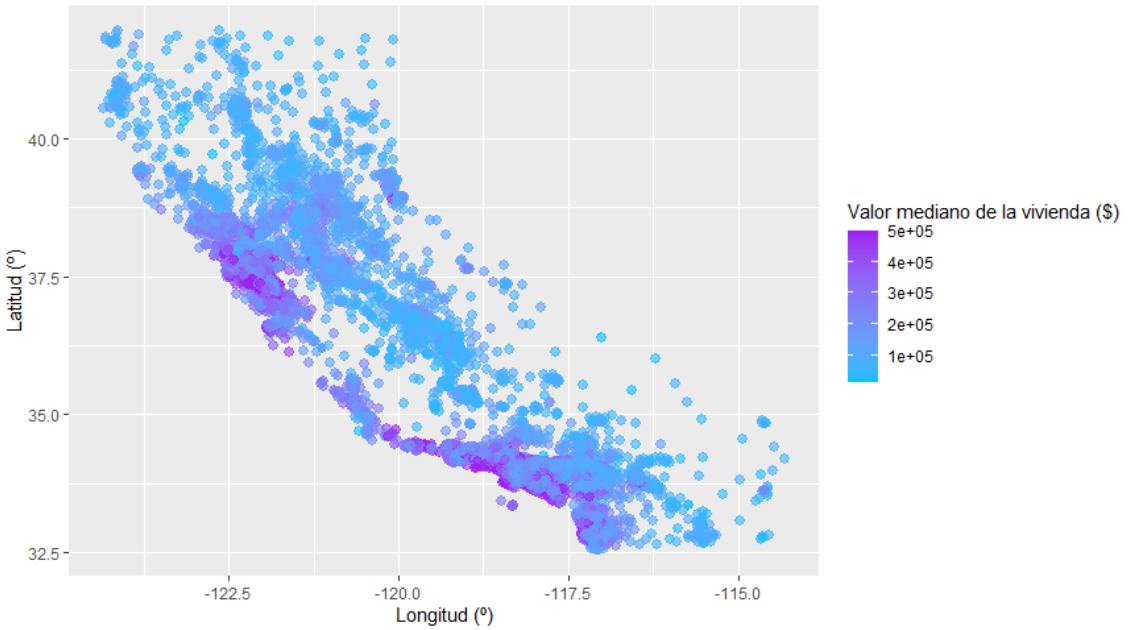

Figura 5: Gráfico que muestra la latitud frente la longitud y su correspondencia con la variable dependiente.

Siguiendo con la segunda hipótesis, es evidente que existen zonas de aglomeración de viviendas en varias zonas. Yendo más allá, parece ser que las viviendas más caras se sitúan en la zona costera del estado de california, lo cual tiene sentido.

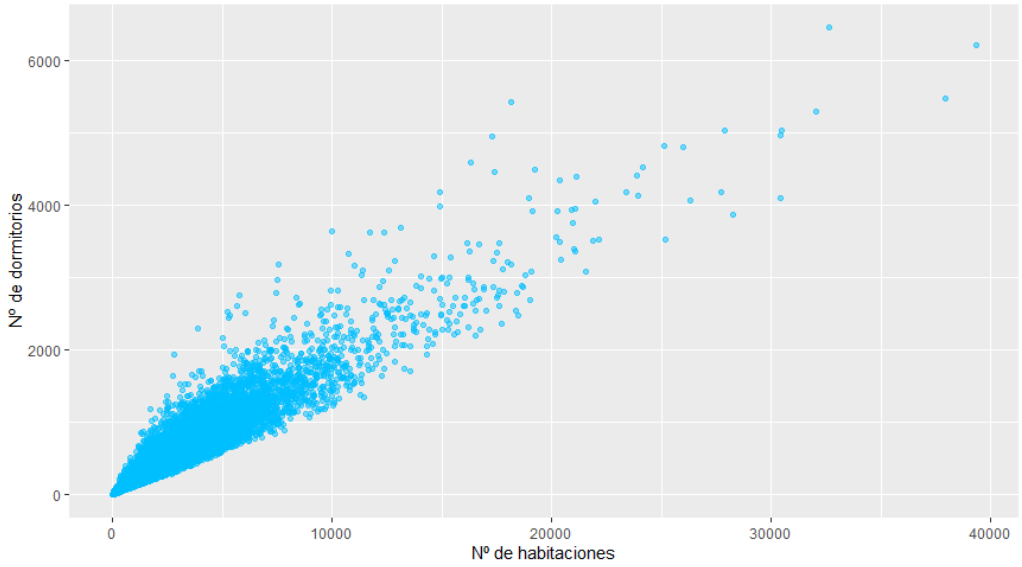

Figura 6: Gráfico que muestra la dependencia entre el número de dormitorios y el número de habitaciones.

La figura 6 muestra como la hipótesis de la dependencia entre el número de dormitorios y el número de habitaciones se cumple.

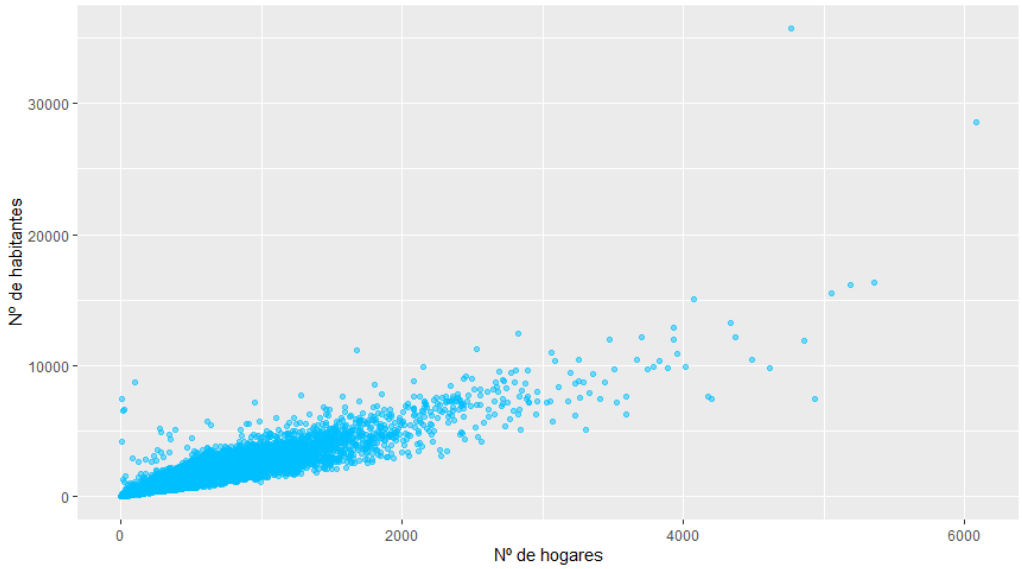

Figura 7: *Scatter plots* bivariables de todos los atributos.

De igual manera que en el caso anterior, la relación entre el número de habitantes de una localidad y el número de hogares queda reflejada en la figura 7.

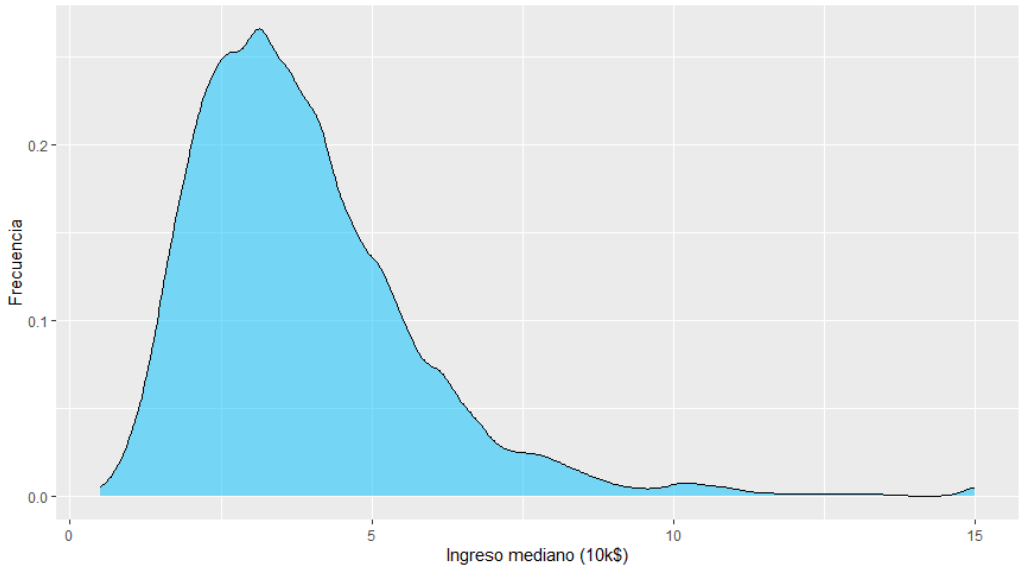

Figura 8: Gráfico que muestra la distribución de la variable MedianIncome.

Por último se ha comprobado que la distribución de los ingresos por localidad está sesgada hacia la derecha, como muestra la figura 8.

1.1.5. Correlación y reducción de la dimensionalidad

En este apartado se estudiará la correlación entre las distintas características y conforme a ello se realizará una reducción de la dimensionalidad de manera conveniente.

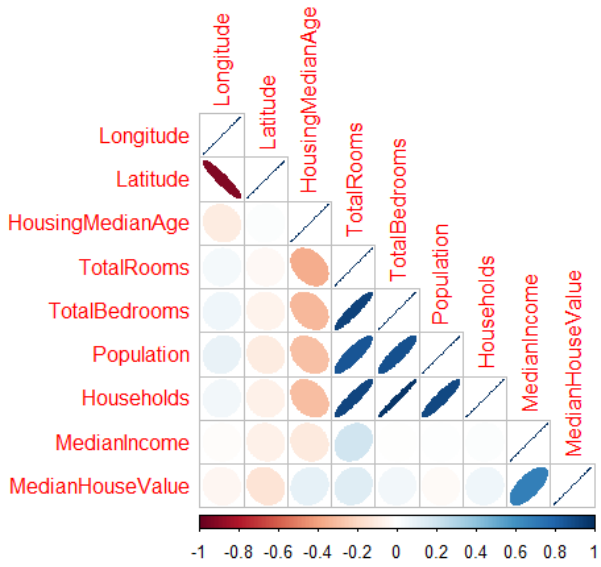

Figura 9: Correlación de las distintas características.

En la figura 9 se puede observar la correlación entre las distintas variables de manera visual. Teniendo en cuenta esto, se puede realizar una reducción de la dimensionalidad para manejar las variables correlacionadas. No conviene eliminar variables correlacionadas, en este caso es preferible crear nuevas variables a partir de antiguas para evitar la pérdida de informiación. Por ejemplo, como Population, TotalBedrooms, TotalRooms y Households están muy correlacionadas entre ellas, se pueden crear nuevas variables del tipo $\frac{X}{Population}$. De la misma manera, Latitude y Longitude están altamente correlacionadas, lo que permite crear una nueva variable que mida la distancia al cuadrado a un punto arbitrario teniendo en cuenta las coordenadas. Dicho punto arbitrario puede ser la media de estas variables.

Esta última variable se crearía de la siguiente forma:

$$DistanciaCentro = (Latitude - mean(Latitude))^2 + (Longitude - mean(Longitude))^2$$

Al realizar estas transformaciones, se cambian las variables **Population**, **TotalBedrooms**, **TotalRooms** y **Households** por **BedroomsPorPopulation**, **RoomsPorPopulation** y **HouseholdPorPopulation**. Por otro lado, las variables **Latitude** y **Longitude** quedarían resumidas en **DistanciaCentro**.

1.1.6. Transformación de los datos

En este último apartado de procesamiento de los datos se va a estudiar la normalidad de las distintas variables. Para ello se mostrará un gráfico de tipo *QQ plot* y se aplicará el test de *Shapiro-Wilks* sobre una muestra del dataset. Dependiendo de ello, se sugerirán dos maneras de transformar los datos y se optará por aquella que mejores resultados proporcione al repetir dicho test.

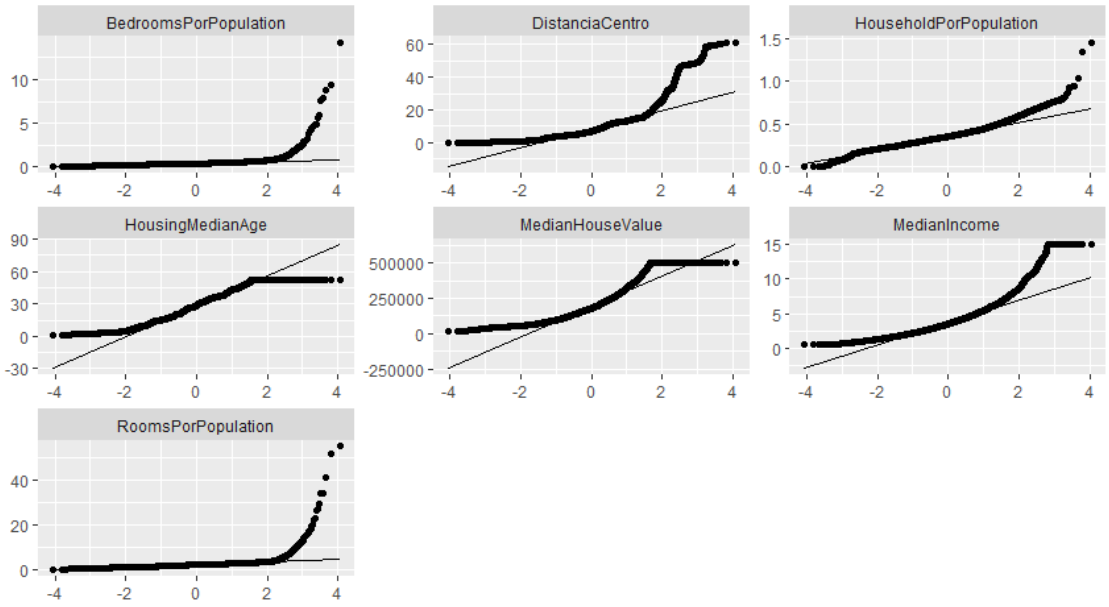

Figura 10: *QQ plots* de las variables.

Continuando con los resultados del test de *Shapiro-Wilk's* cabe aclarar que en dicho test, la hipótesis nula H_0 consiste en asumir que las variables se distribuyen de forma normal, de manera que si $p - valor < 0.05$ se puede asumir que la distribución de los datos es notablemente diferente a la de una normal.

Característica	p-valor Test S-W
DistanciaCentro	$3.5 \cdot 10^{-62}$
HousingMedianAge	$1.8 \cdot 10^{-26}$
RoomsPorPopulation	$6.4 \cdot 10^{-76}$
BedroomsPorPopulation	$9.7 \cdot 10^{-81}$
HouseholdPorPopulation	$1.2 \cdot 10^{-35}$
MedianIncome	$1.76 \cdot 10^{-48}$
MedianHouseValue	$1.0 \cdot 10^{-46}$

Tabla 2: p-valor del test *Shapiro-Wilk's* para todas las variables.

A la luz de los gráficos de la figura 10 y los resultados que se muestran en la tabla 2, se puede concluir que los datos necesitan ser transformados, no siguen una distribución normal y por otro lado, presentan distribuciones sesgadas positivamente.

Se probará a transformar los datos tanto logarítmicamente como también usando el método *Yeo-Johnson* utilizando el paquete *caret*. En ambos casos, posteriormente a la transformación, los datos se estandarizarán.

La transformación *Yeo-Johnson* transforma los datos para cada variable de la siguiente manera:

$$y' = \begin{cases} \left(\frac{(y+1)^{\lambda}-1}{\lambda}\right) & \text{si } y \geq 0, \lambda \neq 0 \\ \ln(y+1) & \text{si } y \geq 0, \lambda = 0 \\ -\left(\frac{(-y+1)^{2-\lambda}-1}{2-\lambda}\right) & \text{si } y < 0, \lambda \neq 2 \\ -\ln(-y+1) & \text{si } y < 0, \lambda = 2 \end{cases}$$

donde λ es un parámetro que se calcula automáticamente utilizando la función *preProcess()* del paquete *caret*.

Característica	p-valor Test S-W (Y-J)	p-valor Test S-W (ln)
DistanciaCentro	$6.8 \cdot 10^{-23}$	$9.5 \cdot 10^{-44}$
HousingMedianAge	$2.1 \cdot 10^{-25}$	$8.5 \cdot 10^{-49}$
RoomsPorPopulation	$1.4 \cdot 10^{-37}$	$4.8 \cdot 10^{-50}$
BedroomsPorPopulation	$6.8 \cdot 10^{-79}$	$4.2 \cdot 10^{-55}$
HouseholdPorPopulation	$2.8 \cdot 10^{-21}$	$6.1 \cdot 10^{-53}$
MedianIncome	0.075	$3.5 \cdot 10^{-7}$
MedianHouseValue	$7.4 \cdot 10^{-20}$	$3.4 \cdot 10^{-22}$

Tabla 3: p-valor del test *Shapiro-Wilk's* para todas las variables tras las transformaciones y estandarización.

Como se puede observar en la tabla 3 la transformación por *Yeo-Johnson* de los datos junto a una estandarización usual de los datos proporciona mejores resultados que la transformación por logaritmo y estandarización. Es más, para la variable **MedianIncome**, el p-valor es mayor que 0.05, por lo que se puede asumir normalidad para esta variable. Este aspecto es bastante positivo, ya que dicha variable es la que más correlacionada está con la variable dependiente.

1.1.7. Conclusiones

Como conclusión de este análisis se puede mencionar como las hipótesis realizadas al principio se han verificado a través del uso de gráficos bivariables y univariables. Por otro lado, se ha llevado a cabo un procesamiento de las variables y los datos de manera que se han reducido el número de características conservando la mayor parte de la información, además de realizar dos transformaciones de los datos, una de ellas es la logarítmica y la otra es por Yeo-Johnson, ambas transformaciones han terminado con un estandarización. La transformación por Yeo-Johnson ha permitido que se pueda asumir que la variable independiente con mayor correlación a la dependiente se distribuya de manera normal, por ello sería recomendable utilizarla para el problema de regresión.

1.2. Bupa

1.2.1. Definición del problema

En el siguiente caso se trabajará con el dataset *bupa*, para el cual se tiene que resolver un problema de clasificación. Este dataset, también creado en 1990 fue recopilado originalmente por la empresa *BUPA Medical Research Ltd.* para estudiar posibles vínculos entre indicadores biológicos, el consumo excesivo de alcohol y enfermedades hepáticas y cuenta con las variables **Mcv, Alkphos, Sgpt, Sgot, Gammagt, Drinks** y **Selector**.

El problema a resolver es crear un modelo de clasificación que pueda predecir correctamente si existe presencia de algún desorden hepático a partir de las demás características, por lo tanto la variable dependiente se trata de **Selector**, las demás son independientes.

Hay que tener en cuenta que en principio, la variable dependiente era **Drinks**, la característica **Selector** era solo un índice que servía para dividir el dataset en conjuntos de entrenamiento y test. En estudios posteriores se ha malinterpretado y se utiliza como indicador sobre la presencia o no de enfermedad hepática. Se puede encontrar más información en el siguiente artículo: https://www.sciencedirect.com/science/article/pii/S0167865516000088#tbl0001. Sea como sea, se usará la variable **Selector** como característica a predecir.

Las hipótesis realizadas para este problema son las siguiente:

1. La variable **Drinks** estará relacionada con la presencia o no de desórden hepático.
2. Las características correspondientes a indicadores biológicos presentarán alguna relación entre ellas.
3. Las distribuciones de los indicadores biológicos tomarán una forma parecida a una distribución normal (con sesgo probablemente).

Más adelante, con el uso de gráficos, podremos verificar si estas hipótesis son acertadas y se procesarán los datos de manera conveniente conforme a ellas.

1.2.2. Descripción de los datos

El dataset presenta formato *csv* separado por comas, para leerlo como un *data frame* se puede usar la función *read.csv*. Este contiene **345** instancias y **7** características que se han mencionado ya anteriormente. Los tipos de datos atómicos que se tienen son:

1. **Mcv**: Volumen corpuscular medio en femtolitros, tipo *integer* (entero).
2. **Alkphos**: Fosfatasa alcalina en unidades por litro de sangre, tipo *integer* (entero).
3. **Sgpt**: Alanina aminotransferasa (ALT) en unidades por litro, tipo *integer* (entero).
4. **Sgot**: Aspartato aminotransferasa (AST) en unidades por litro, tipo *integer* (entero).
5. **Gammagt**: Gamma-glutamil transferasa en unidades por litro, tipo *integer* (entero).
6. **Drinks**: Consumo de alcohol diario en unidades de media pinta, tipo *numeric* (real).
7. **Selector**: Presencia (1) o ausencia (2) de desorden hepático, tipo *factor* (categórico).

Por otro lado, podemos hacernos una idea del contenido del dataset realizando un rápido estudio estadístico de las distintas variables cuantitativas, que se presenta a continuación en la tabla 4.

Característica	Media	Mediana	Mínimo	Máximo	1er Cuartil	3er Cuartil	Desv. Est.
Mcv	90.16	90.00	65.00	103.00	87.00	93.00	4.45
Alkphos	69.87	67.00	23.00	138.00	57.00	80.00	18.30
Sgpt	30.41	26.00	4.00	155.00	19.00	34.00	19.50
Sgot	24.64	23.00	5.00	82.00	19.00	27.00	10.10
Gammagt	38.28	25.00	5.00	297.00	15.00	46.00	39.30
Drinks	3.46	3.00	0.00	20.00	0.50	6.00	3.34

Tabla 4: Estudio estadístico de las características cuantitativas de *bupa*.

Por otro lado, en la tabla 5 encontramos el recuento de las clases de la variable dependiente **Selector**.

	Presencia de Desord. Hepático (1)	Ausencia de Desord. Hepático (2)
Cantidad	145	200

Tabla 5: Tabla de la variable dependiente (categórica) de *bupa*.

1.2.3. Procesamiento de los datos

Primero antes de todo, es necesario transformar los tipos de datos de algunas variables, ya que al importar el dataset como *data frame* las características toman un tipo incorrecto.

En concreto se han realizado las siguientes transformaciones de tipos:

- Las variables **Mcv**, **Alkphos**, **Sgpt**, **Sgot** y **Gammagt** se han forzado a tomar el tipo *integer* (entero), ya que se importaron como reales.
- La variable dependiente **Selector** se ha transformado a tipo *factor* (categórico) ya que se importó como tipo *integer* (entero), además se han cambiado las etiquetas de 1 y 2 (Presencia/Ausencia de desorden hepático) a una binaria usual de 1 y 0 (Presencia/Ausencia de desorden hepático).

Por otro lado, el dataset no presenta valores perdidos en ninguna de sus características, por lo tanto en ese sentido no es necesario realizar ningún tipo de imputación ni eliminación de variables o instancias.

En cuanto a los *outliers*, podemos identificarlos utilizando un gráfico de tipo *box plot* para cada variable. Este se muestra a continuación en la figura 11.

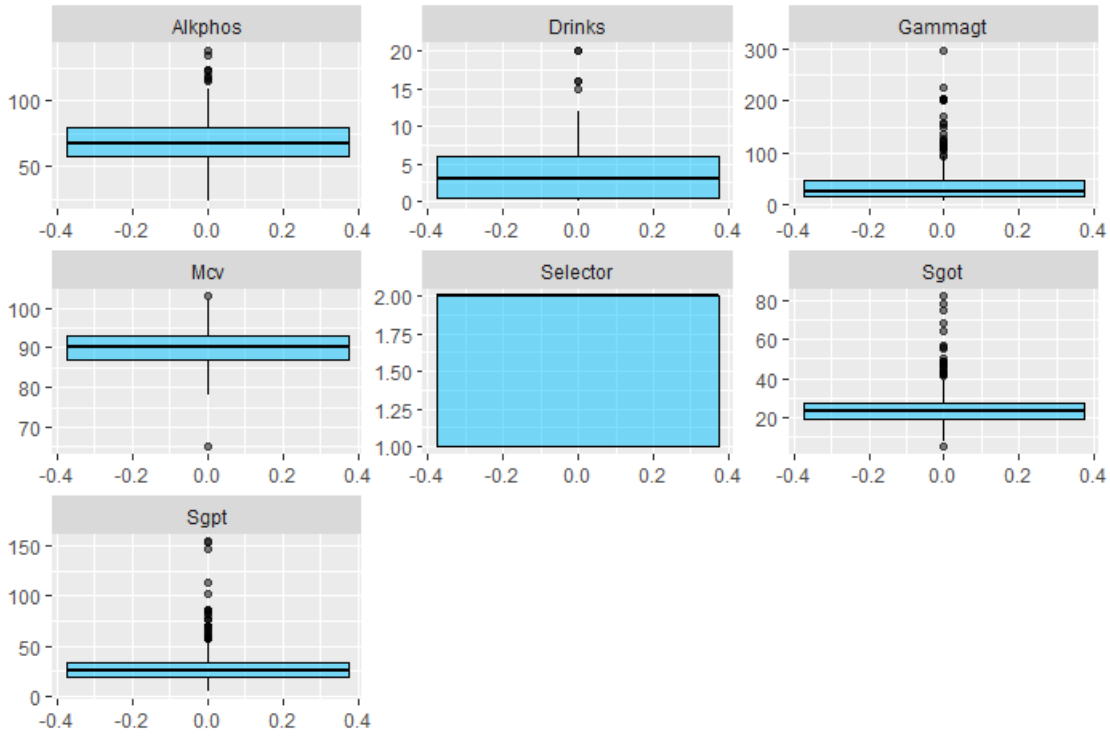

Figura 11: Distribución de datos para cada variable usando *box plots*.

Como se puede observar en la figura anterior, todas las variables independientes presentan *outliers*. Sin embargo, al igual que en el caso del anterior dataset, se ha optado por conservarlos debido a que se tratan de *outliers* multivariables cuyas causas son de tipo natural, por lo tanto son necesarios para que el modelo pueda captar adecuadamente toda la variabilidad que presenta la muestra y poder clasificar correctamente las instancias.

1.2.4. Resumen de los datos

A continuación, se muestran gráficos univariables y bivariables para tener una perspectiva global del conjunto de datos con el que se está trabajando y ver como se relacionan las distintas características, así como otros gráficos más concretos que servirán para verificar las distintas hipótesis que se han hecho anteriormente.

En la figura 12 encontramos los *plots* univariables, mientras que en la figura 13 se muestran los gráficos bivariables.

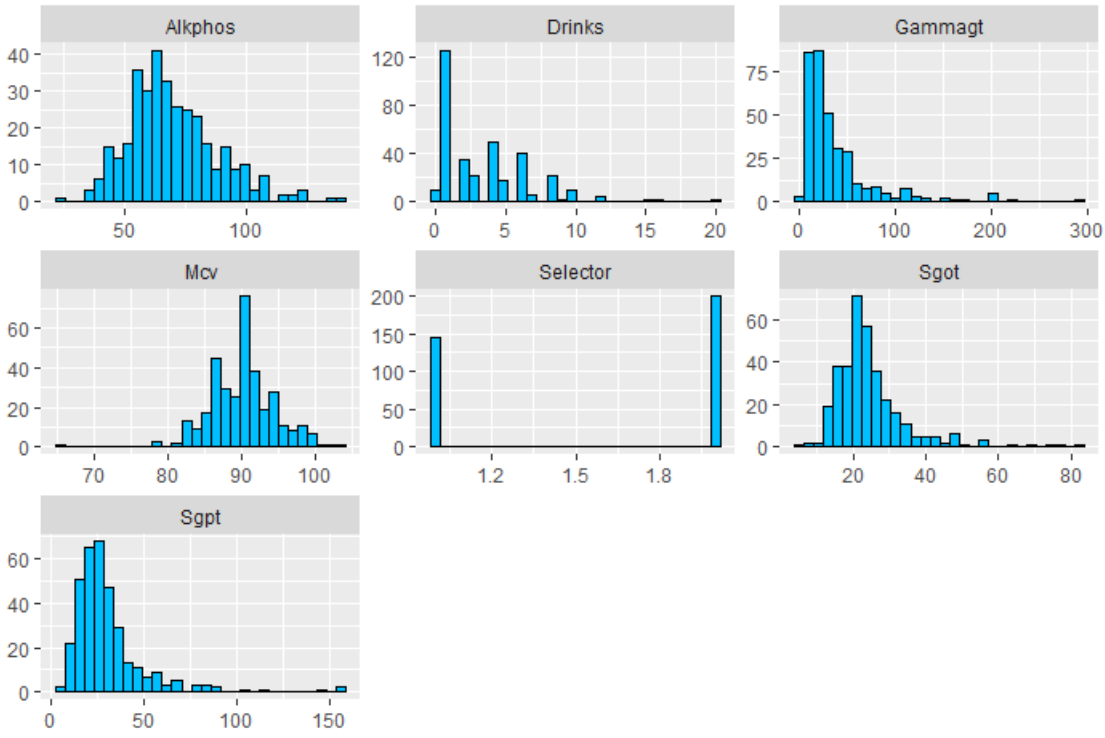

Figura 12: Histogramas de todas los atributos por separado.

Por otro lado, los plots bivariables se muestran en la figura 13.

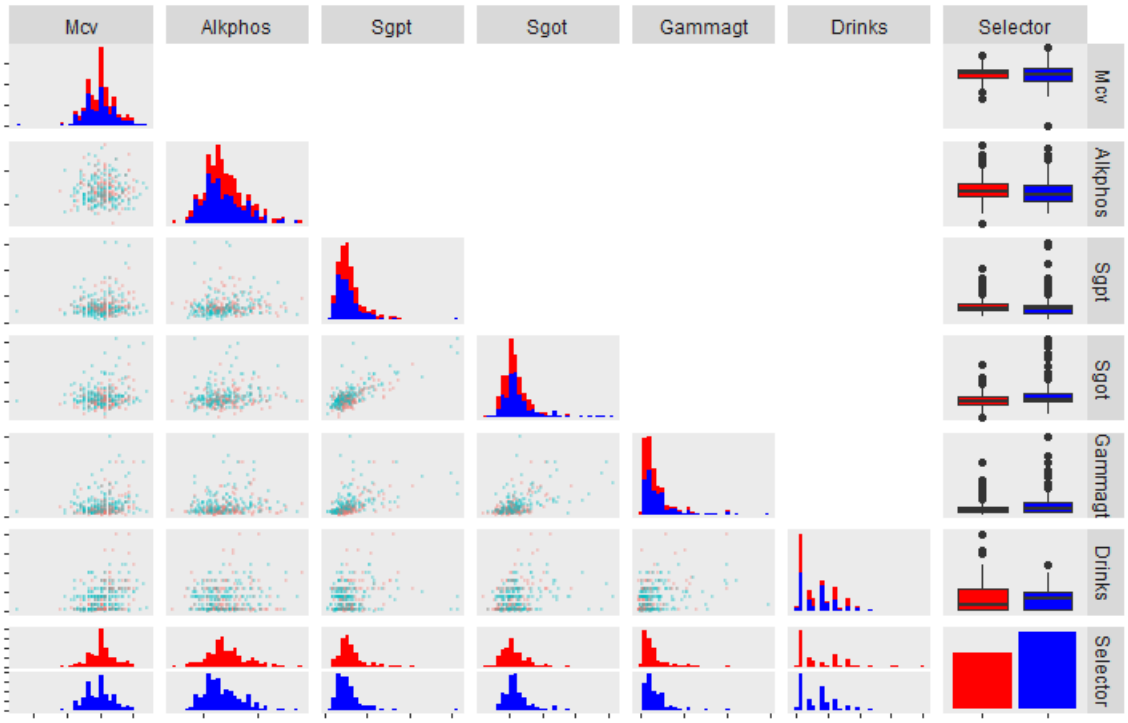

Figura 13: *Scatter plots* bivariables de todos los atributos.

Igual que para el caso de *california*, en la figura 13 se ha optado por eliminar las escalas de los ejes, ya que el propósito principal de este gráfico es tener una perspectiva general sobre cómo se relacionan las características en parejas. Aquí nos encontramos con que aparentemente no existe ninguna relación evidente entre las variables, pero eso es un aspecto que trataremos en el siguiente apartado, donde lo estudiaremos más detenidamente. En lo que respecta a la figura 12, podemos ver que tipo sesgo y curtosis tienen las variables. Algunas variables como **Gammagt**, **Sgot** y **Sgpt** tienen una alta curtosis y están sesgadas hacia la derecha. Por otro lado, características como **Alkphos** y **Mcv** parecen tener una curtosis más moderada y un sesgo también menos pronunciado.

En otro orden de cosas, a continuación se muestran gráficos relativos a las hipótesis que se han hecho cuando hemos definido este segundo problema.

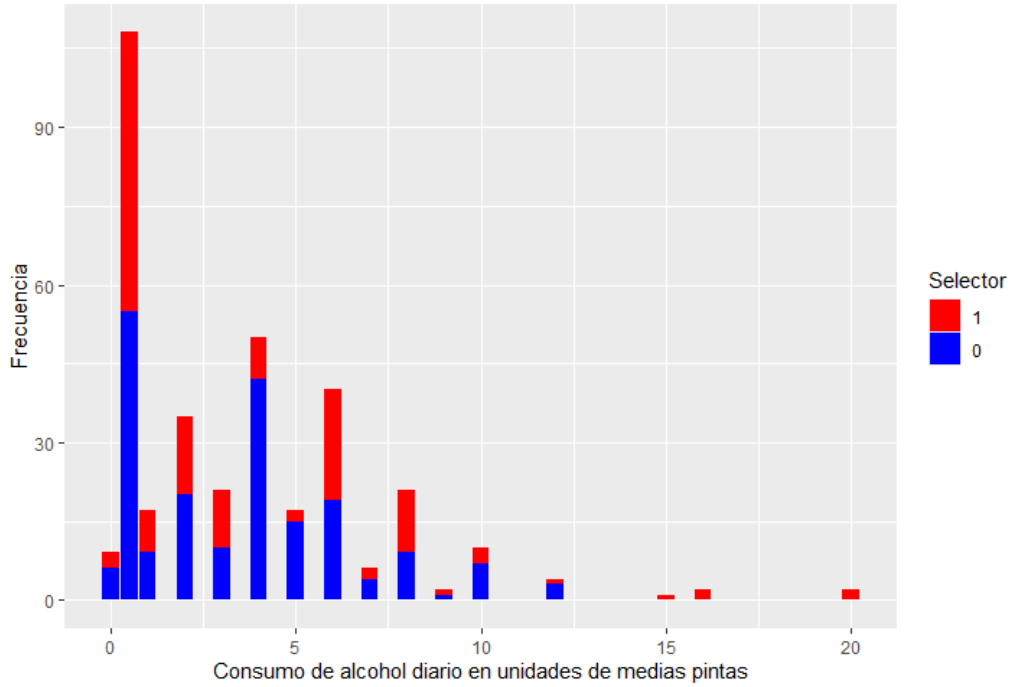

Figura 14: Gráfico que muestra la frecuencia de **Drinks** facetado según la variable dependiente **Selector**.

La primera hipótesis que se ha realizado es que existe relación entre el consumo diario de alcohol y la presencia de algún tipo de enfermedad en el hígado. Como se puede observar en la figura 14, esta hipótesis no se confirma, no parece existir relación alguna entre estas variables, exceptuando casos de *outliers* por encima de las 14 unidades de medias pintas diarias. Estas instancias pertenecen a las clase 1. Este hecho se usará posteriormente para crear una variable **Drinks** más útil.

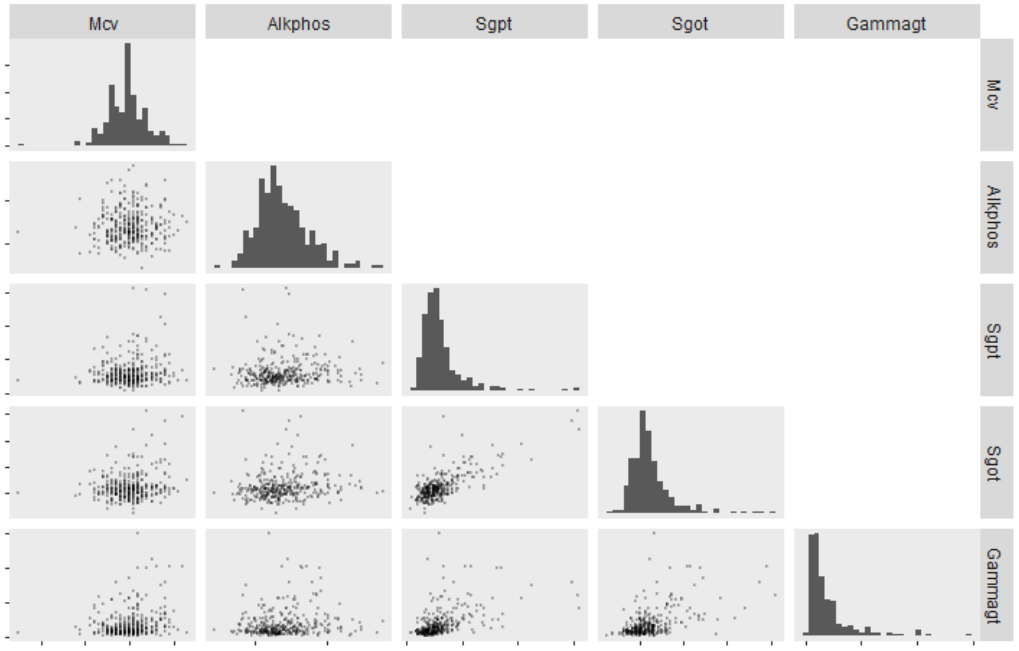

Figura 15: Gráficos bivariables de las caracterísitcas que son indicadores biológicos.

La segunda hipótesis queda representada en la figura 15. Si bien la mayoría de estos gráficos no presentan evidencia de correlación alguna, fijándonos en el gráfico bivariable de **Sgot** y **Sgpt** se puede observar que es probable que exista cierta correlación entre estas características. La figura 16 muestra este gráfico en detalle, confirmando esta segunda hipótesis.

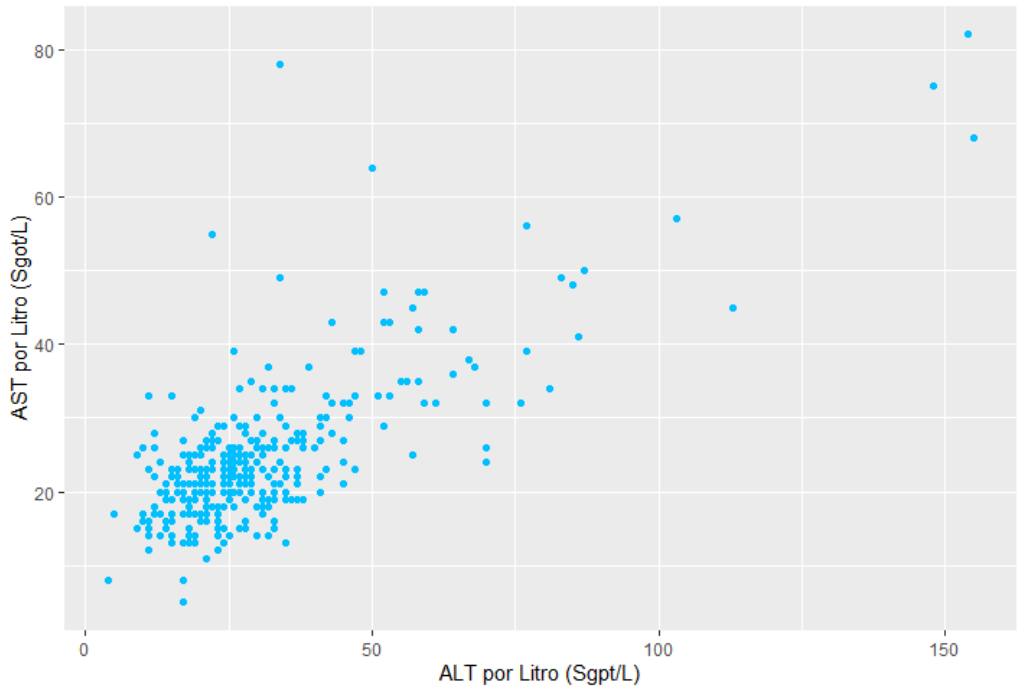

Figura 16: Gráfico que muestra el AST frente al ALT por Litro.

Por último, queda demostrar la última hipótesis acerca de la distribución de estas características sobre indicadores biológicos. Las densidades de cada distribución quedan reflejadas en la figura 17.

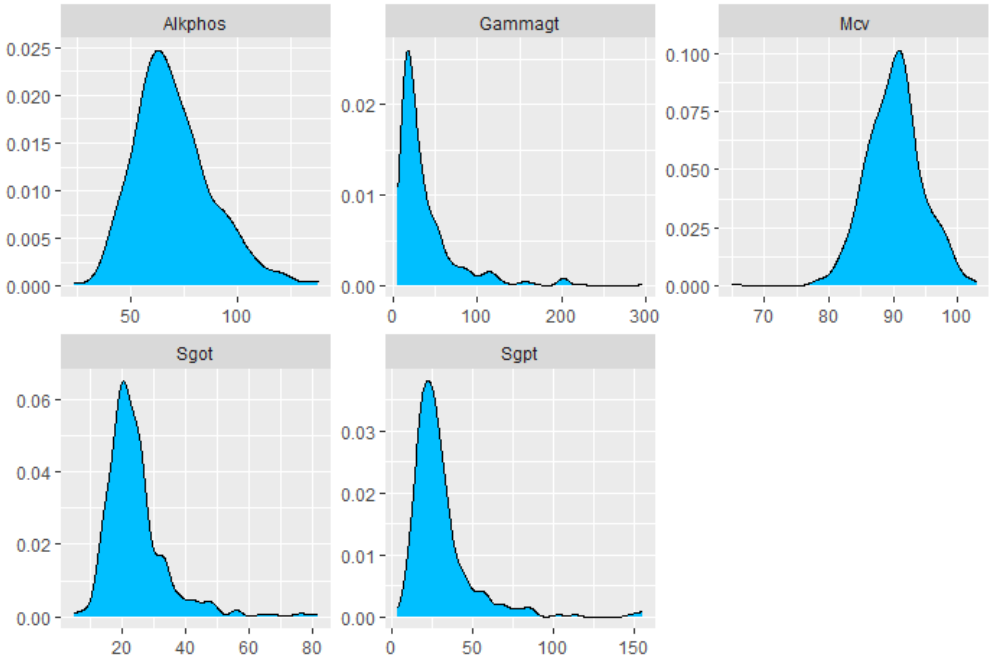

Figura 17: Gráfico que muestra las densidades de distribución de las variables de indicadores biológicos.

Variables como **Alkphos** y **Mcv** presentan una densidad de distribución que se asemeja en cierta medida a una distribución normal.

1.2.5. Correlación y reducción de dimensionalidad

En este apartado se estudiará la correlación entre las distintas características y conforme a ello se realizará una reducción de la dimensionalidad y modificación de variables, utilizando alguna de las hipótesis como motivo.

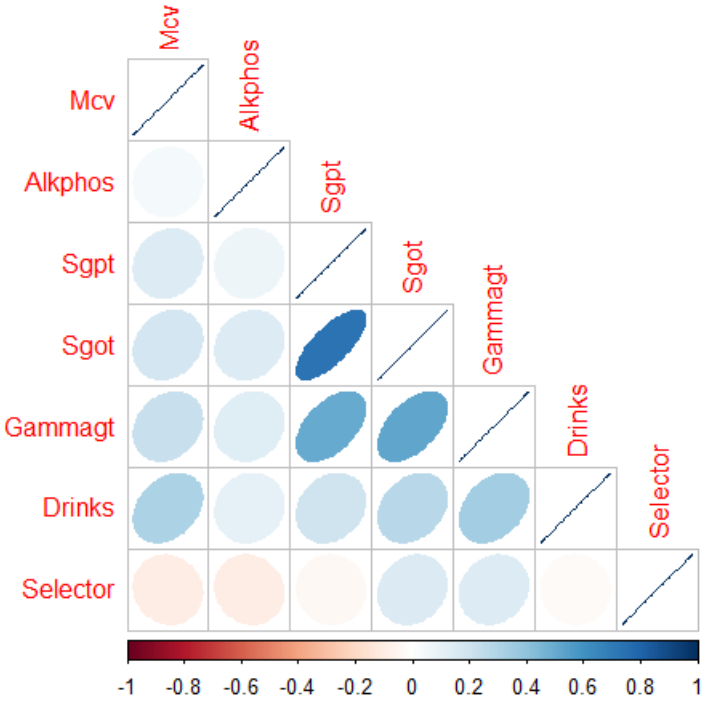

Figura 18: Correlación de las distintas características.

Como se explicó en el apartado anterior y se puede ver en la figura 18, las variables **Sgot** y **Sgpt** presentan una alta correlación (y un poco **Gammagt**) entre ellas. En esta situación aplicar una reducción por PCA no es factible, ya que tenemos pocas variables, y solo existe una alta correlación entre dos de ellas. En vez de esto podemos crear una nueva variable basándonos en la proporción $\frac{AST}{ALT}$ ($\frac{\textbf{Sgot}}{\textbf{Sgpt}}$). Este índice si es menor a 1 está relacionado con la existencia de hepatitis crónica y síndromes colestásicos crónicos. Se puede encontrar más información sobre esto en el siguiente enlace: https://www.gastrojournal.org/article/S0016-5085(88)80022-2/fulltext.

El recuento de clases según el valor de este índice queda representado en la tabla 6.

Selector	$\frac{AST}{ALT} < 1$	$\frac{AST}{ALT} > 1$
Presencia de desorden	113	25
Ausencia de desorden	105	86

Tabla 6: **Selector** en función de $\frac{AST}{ALT}$

Como vemos, aporta algo de información sobre la variable dependiente. Por ello se creará una variable categórica binaria que tendrá un valor de 1 si $\frac{AST}{ALT} < 1$ y 0 en caso contrario. Esta variable sustituirá a las variables **Sgpt** y **Sgot** que por sí mismas no nos aportan demasiada información acerca de la variable dependiente.

Otro cambio que se va a realizar concierne a la variable **Drinks**. Aparentemente no existe correlación alguna entre esta característica y la variable a predecir pero observando la figura 14 podemos deducir que para consumos de mayores a 15 medias pintas diarias esto no es así, ya que este pequeño grupo de datos pertenece a la clase que indica la presencia de algún desorden hepático. Por ello, se sustituirá la variable **Drinks** por otra variable categórica que tomará 1 como valor si **Drinks** ≥ 15 y 0 en caso contrario.

1.2.6. Transformación de los datos

En este último apartado de procesamiento de los datos se va a estudiar la normalidad de las distintas variables numéricas importantes (incluso las que se han eliminado por reducción de la dimensionalidad). Para ello se mostrará un gráfico de tipo *QQ plot* y se aplicará el test de *Shapiro-Wilks* sobre todas las instancias. Posteriormente se aplicará una transformación logarítmica y luego una estandarización usual para estas variables numéricas.

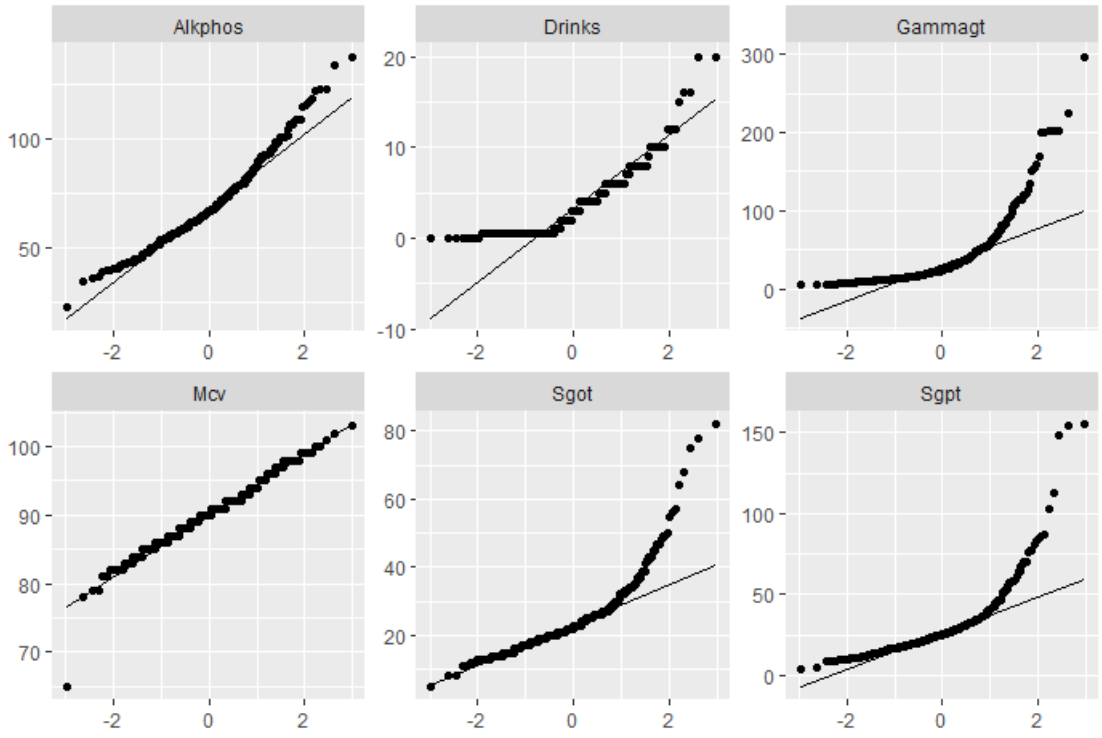

Figura 19: *QQ plots* de las variables numéricas.

Los resultados del test de *Shapiro-Wilk's* se encuentran a continuación, en la tabla 7.

Característica	p-valor Test S-W
Mcv	$3.3 \cdot 10^{-6}$
Alkphos	$3.6 \cdot 10^{-7}$
Sgpt	$2.6 \cdot 10^{-23}$
Sgot	$1.4 \cdot 10^{-19}$
Gammagt	$6.5 \cdot 10^{-25}$
Drinks	$1.7 \cdot 10^{-18}$

Tabla 7: p-valor del test *Shapiro-Wilk's* para las variables numéricas.

A la luz de los gráficos de la figura 19 y los resultados que se muestran en la tabla 7, se puede concluir que los datos necesitan ser transformados, no siguen una distribución normal y por otro lado, exceptuando la variable **Mcv**, presentan distribuciones sesgadas positivamente.

Se probará a transformar los datos logarítmicamente (menos la variable **Mcv**). Posteriormente a la transformación, todos los datos numéricos se estandarizarán.

A continuación encontramos los p-valores del test de *Shapiro-Wilk's* tras la transformación y la estandarización.

Característica	p-valor Test S-W
Mcv	$3.3 \cdot 10^{-6}$
Alkphos	0.25
Sgpt	$1.5 \cdot 10^{-5}$
Sgot	$2.7 \cdot 10^{-7}$
Gammagt	$3.8 \cdot 10^{-6}$
Drinks	$8.7 \cdot 10^{-14}$

Tabla 8: p-valor del test *Shapiro-Wilk's* para las variables numéricas depués de la transformación y estandarización.

Como se puede observar en la tabla 8 la transformación logarítmica de los datos junto a una estandarización usual mejora los resultados del test *Shapiro-Wilk's*. Es más, para la variable **Alkphos**, el p-valor es bastante mayor que 0.05, por lo que se puede asumir normalidad para esta variable. Esto tendrá un efecto positivo en la implementación de algoritmos de clasificación como LDA y QDA, ya que se ven afectados negativamente si los datos se desvían mucho de una distribución normal.

1.2.7. Conclusiones

Finalmente como conclusión se puede comentar la gran dificultad que presenta este dataset en cuanto a crear un modelo de clasificación para el mismo. A raíz de que la primera hipótesis haya quedado desmentida, se ha mostrado que ninguna característica presenta correlación relevante con la variable dependiente, y por otro lado las variables independientes no se prestan a ser reducidas dimensionalmente. Sin embargo, realizando las transformaciones adecuadas se ha conseguido que una variable se distribuya de manera normal.

En cuanto a las variables que se podrían utilizar para el problema de clasificación, estas podrían variar dependiendo del algoritmo que se aplique. Una cosa es segura y es que los algoritmos LDA y QDA manejan únicamente variables numéricas, por lo tanto se optará por utilizar el dataset

transformado pero sin la reducción de la dimensionalidad y modificación propuestas en el capítulo 1.2.5. Por otro lado se tendrán que comprobar los resultados que brinda el algoritmo k-NN para ambos casos, ya que este sí es capaz de manejar variables categóricas (aunque se tendrán que convertir a numéricas para que las pueda procesar); formar un juicio a priori en esta situación es imposible.

2. Trabajo sobre Regresión

En esta sección se aborda el problema de regresión propuesto para el dataset *california*. Para ello, se han seguido las indicaciones de las sesiones de laboratorio realizadas en clase.

2.1. Regresión lineal simple

En este apartado se ha realizado una regresión lineal simple sobre 5 de los 8 regresores. El criterio que se ha seguido para seleccionar estos 5 regresores ha consistido en realizar un análisis de correlación con la variable dependiente y escoger las características con mayor correlación. Estas 5 características han sido: **MedianIncome**, **Latitude**, **TotalRooms**, **HousingMedianAge** y **Households**.

A continuación, en la figura 20 se muestran los distintos ajustes para cada variable.

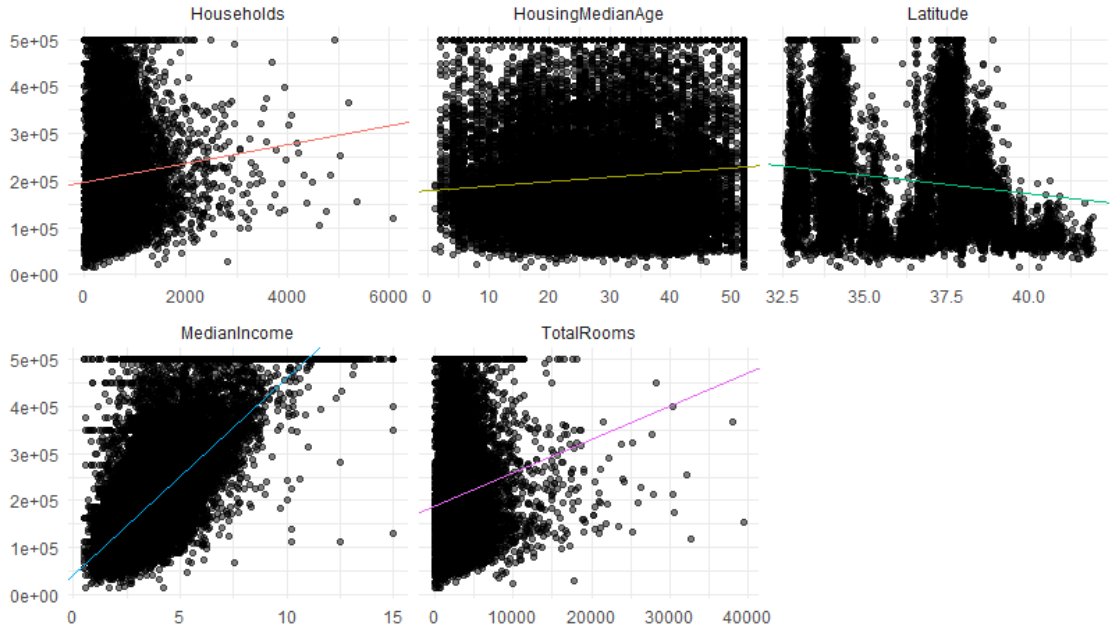

Figura 20: Regresiones lineales para las 5 variables independientes con mayor correlación.

Además, se ha creado un *dataframe* con los valores del coeficiente R^2 y de la raíz del error

cuadrático medio (RMSE) de cada regresión. La información de este dataframe queda reflejada en la tabla 9.

Característica	R^2	$RMSE$
MedianIncome	0.473	583
Latitude	0.021	795
TotalRooms	0.018	796
HousingMedianAge	0.011	799
Households	0.004	801

Tabla 9: Valores R^2 y $RMSE$ de las características con mayor correlación con la variable dependiente.

A la luz de los resultados obtenidos, es evidente que el mejor modelo de regresión lineal simple es el que utiliza **MedianIncome** como variable predictora, que es además la variable que más correlación presenta con la variable independiente. En la figura 21 podemos observar este modelo con mayor detalle.

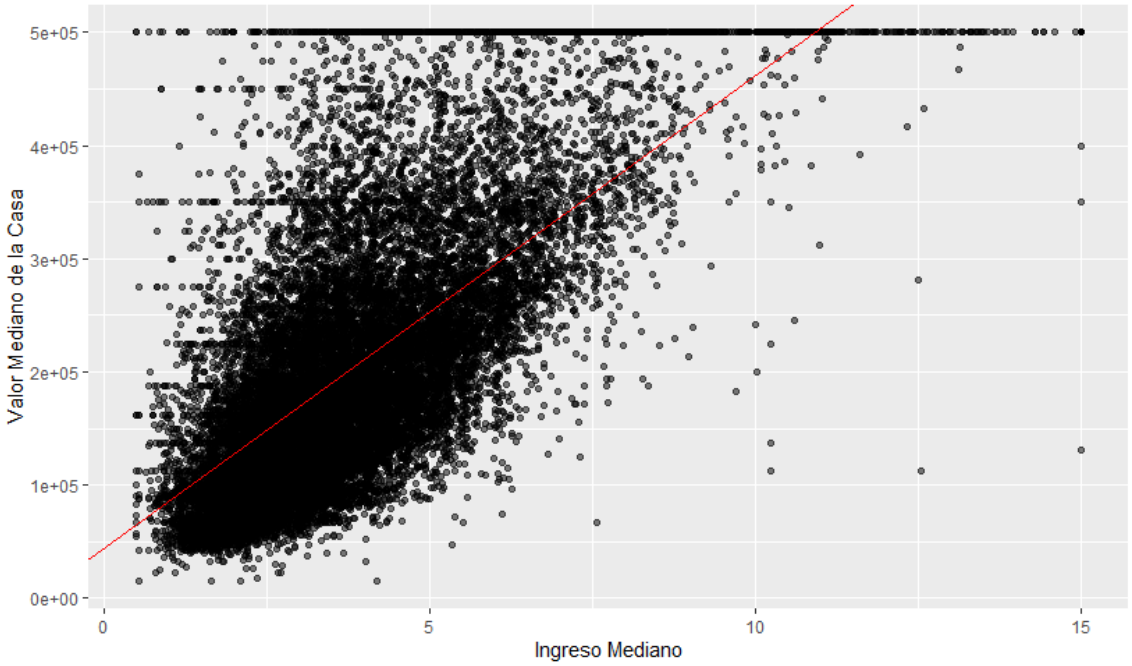

Figura 21: Mejor modelo de regresión lineal simple encontrado.

2.2. Regresión múltiple

En este apartado se ha estudiado una serie de modelos de regresión tanto lineales como no lineales múltiples, es decir, usando más de una variable predictora.

En primer lugar se ha realizado el modelo lineal, que involucra a todas las características. Para este modelo se ha obtenido un valor de 0.637 para el coeficiente R^2 y un $RMSE$ de 484.

Posteriormente, se ha encontrado un modelo de regresión multiple considerando no linealidad e interacciones usando el método de *backward selection*. Para no presentar mucha complejidad, se ha restringido el grado de los términos a grado cuadrático, tanto como para los no lineales como para los de interacciones. Además el *threshold* del p-valor escogido para la eliminación de términos ha sido $< 2 \cdot 10^{-16}$. Es un valor extremadamente restrictivo, pero esto es necesario para equilibrar un modelo no lineal en cuanto a efectividad e interpretabilidad. Para seguir el proceso realizado es mejor consultar el apéndice de esta sección que se encuentra al final del documento, en el que se encuentra el código del proceso realizado y del modelo encontrado, ya que se han aplicado 16 iteraciones del método *backward selection*.

En resumen, el modelo final presenta un coeficiente R^2 de 0.686 y un $RMSE$ igual a 450, mejorando la efectividad del modelo lineal múltiple.

Finalmente, es evidente que estos modelos mejoran con creces el rendimiento del modelo de regresión lineal simple del anterior apartado, los modelos de regresión múltiple permiten modelar relaciones complejas subyacentes en los datos y brindar una mayor flexibilidad a la hora de crear modelos. Sin embargo, presentan una serie de inconvenientes como el sobreaprendizaje y el riesgo de crear un modelo que sea poco interpretable.

2.3. k-NN para regresión

En este apartado se han utilizado las particiones 5-fcv para aplicar el algoritmo de k-NN para regresión y se ha comparado con el modelo no lineal obtenido en el apartado anterior usando las mismas particiones. Para ello se han seguido las directrices presentes en las transparencias de las sesiones de laboratorio.

La medida de calidad utilizada para comparar el rendimiento de ambos modelos ha sido el error medio cuadrático (MSE). En la tabla 10 se muestra la media de este indicador para cada algoritmo y conjunto (entrenamiento y test).

Modelo	Conjunto	MSE
k-NN	Entrenamiento	1562164927
k-NN	Test	3847242396
No lineal	Entrenamiento	4178968671
No lineal	Test	4511996923

Tabla 10: Valor del MSE para los modelos de regresión k-NN y no lineal.

En términos de rendimiento, se observa que el modelo k-NN presenta un MSE mucho menor en el conjunto de entrenamiento comparado con el modelo no lineal. Sin embargo, es notable que la diferencia en los valores de MSE entre los modelos es significativamente mayor en el conjunto de entrenamiento que en el conjunto de test. En este conjunto, aunque el k-NN todavía tiene un MSE más bajo (3847242396) en comparación con el modelo no lineal (4511996923), la diferencia entre los dos modelos es menor en comparación con la observada en el conjunto de entrenamiento. Esta reducción en la diferencia de MSE sugiere que el modelo no lineal, a pesar de tener un rendimiento inferior en general, presenta un menor sobreajuste a los datos de entrenamiento que el algoritmo de k-NN.

2.4. Comparación de los algoritmos

Al igual que en el apartado anterior, esta parte se ha realizado siguiendo las transparencias de las sesiones de laboratorio.

Se han utilizado las particiones 5-fcv para el modelo de regresión por k-NN (realizado en el anterior apartado) y para el modelo lineal múltiple del apartado 2.2 obteniendo los valores del MSE en cada caso. Posteriormente se ha modificado el contenido de los archivos *regr_test_alumnos.csv* y *regr_train_alumnos.csv* para *california* y se han llevado a cabo dos comparaciones. Una comparación ha involucrado los modelos de k-NN y lineal múltiple (tomando el de k-NN como referencia) utilizando el test de *Wilcoxon* y otra comparación además de involucrar estos modelos, incluye también la del algoritmo M5' y se ha realizado usando Friedman y Holm. La hipótesis nula en estos tests consiste en asumir que no existen diferencias significativas entre los distintos modelos.

En la tabla 11 se encuentran los resultados de la comparación por *Wilcoxon* tanto para el conjunto de entrenamiento como para el de test.

Conjunto	R+ (lm)	R- (k-NN)	p-valor
Test	78	93	0.7660
Entrenamiento	10	161	0.0003

Tabla 11: Test de *Wilcoxon* para los modelos de regresión k-NN y lineal múltiple.

En el caso del conjunto de test el p-valor es alto, por lo que podemos tomar la hipótesis nula como cierta, no existen diferencias significativas entre los algoritmos k-NN y lineal múltiple. Sin embargo para el conjunto de entrenamiento, el p-valor es menor a 0.05 por lo que existen diferencias entre los algoritmos como hemos visto en el anterior apartado debido al sobreaprendizaje del modelo k-NN.

Por otro lado, en la tabla 12 podemos ver los resultados del test de *Friedman*.

Conjunto	p-valor
Test	0.015
Entrenamiento	$4 \cdot 10^{-5}$

Tabla 12: Test de *Friedman* para los modelos de regresión k-NN y lineal múltiple.

Para ambos conjuntos se tiene un p-valor bajo, lo que quiere decir que existen diferencias significativas entre 2 algoritmos al menos.

Por último en la tabla 13 se pueden observar los p-valores del test de *Holm* (aplicado post-hoc).

Conjunto	lm vs k-NN	M5' vs lm	M5' vs k-NN
Test	0.58	0.082	0.108
Entrenamiento	0.0031	0.0032	0.0032

Tabla 13: Test de *Holms* para los modelos de regresión k-NN, lineal múltiple y M5'.

A la luz de los resultados de este test, se puede decir que el algoritmo M5' presenta diferencias significativas con los modelos de regresión k-NN y lineal múltiple. En el caso del conjunto test hemos comprobado que entre el modelo k-NN y el lineal múltiple no hay diferencias importantes,

mientras que para el conjunto de entrenamiento sí que las hay debido al sobreaprendizaje explicado anteriormente.

3. Trabajo sobre Clasificación

En esta última sección se encuentra el trabajo propuesto para la parte de clasificación. Se utilizará el dataset *bupa* para ello.

3.1. k-NN

En esta primera parte del trabajo se pide utilizar el algoritmo de clasificación *k-Nearest Neighbours* para el dataset *bupa* en este caso.

Para ello se ha utilizado la función *knn* del paquete *class*. Además, se ha tomado la precisión (exactitud técnicamente) como medida de rendimiento. Por otro lado, el cálculo de este indicador se ha realizado usando la media de los distintos valores de precisión obtenidos utilizando validación cruzada con 20 conjuntos (10 de entrenamiento y 10 de test).

Así mismo, se ha estudiado cuál es el valor de k que mejor rendimiento proporciona en el rango $[1, 20]$ tras haber realizado el preprocesamiento conveniente (transformación logarítmica y estandarización). A continuación en la figura 22 se muestra un gráfico detallado de la precisión del modelo a medida que aumenta el valor de k.

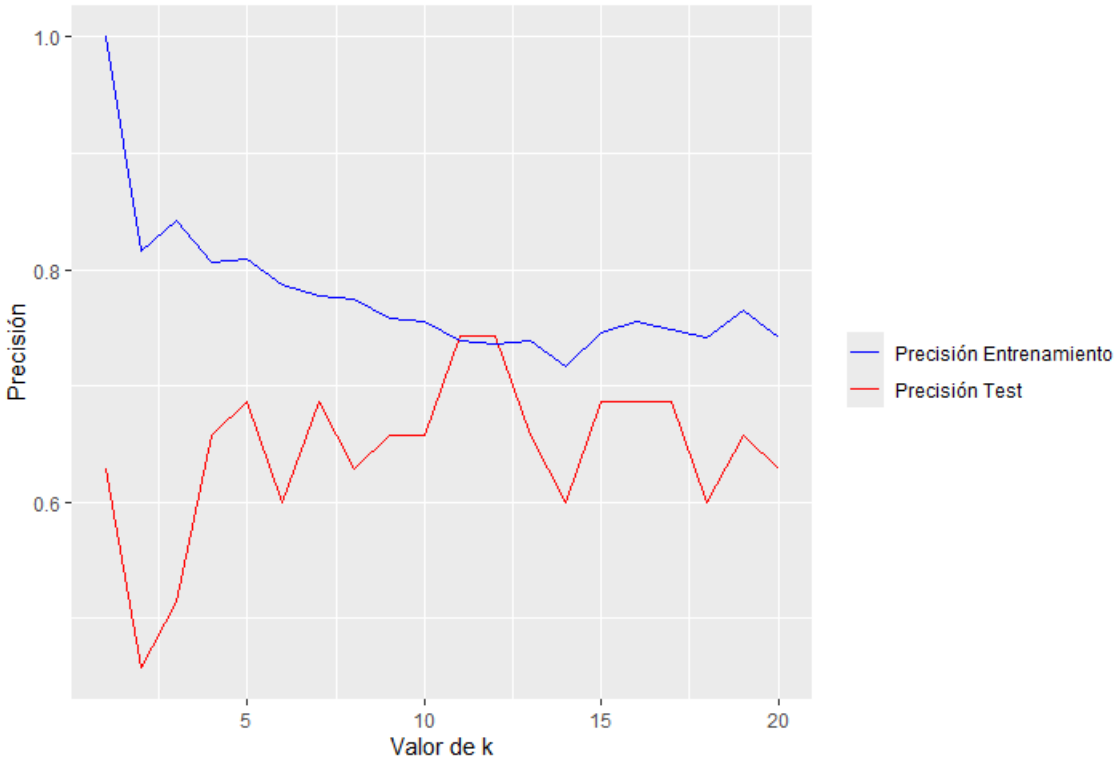

Figura 22: Precisión de k-NN para distintos valores de k, en los conjuntos de entrenamiento y test.

Como se puede observar, es lógico que para $k = 1$ se tenga un rendimiento del 100 % para el conjunto de entrenamiento, ya que como este algoritmo clasifica las instancias (puntos) buscando el vecino más cercano en el espacio de características, si el punto de prueba es el mismo punto que está siendo clasificado, el vecino más cercano de un punto es él mismo y no hay fallos en las predicciones. Dejando esta observación a un lado, parece ser que para $k = \{11, 12\}$ se tiene un rendimiento óptimo en el conjunto de test, incluso un poco por encima del rendimiento para el conjunto de entrenamiento.

Para un modelo con $k = 11$ se tiene el rendimiento que se muestra en la tabla 14 y reflejado en la figura 23.

Conjunto	Rendimiento
Test	0.743
Entrenamiento	0.739

Tabla 14: Rendimiento para el mejor modelo k-NN encontrado.

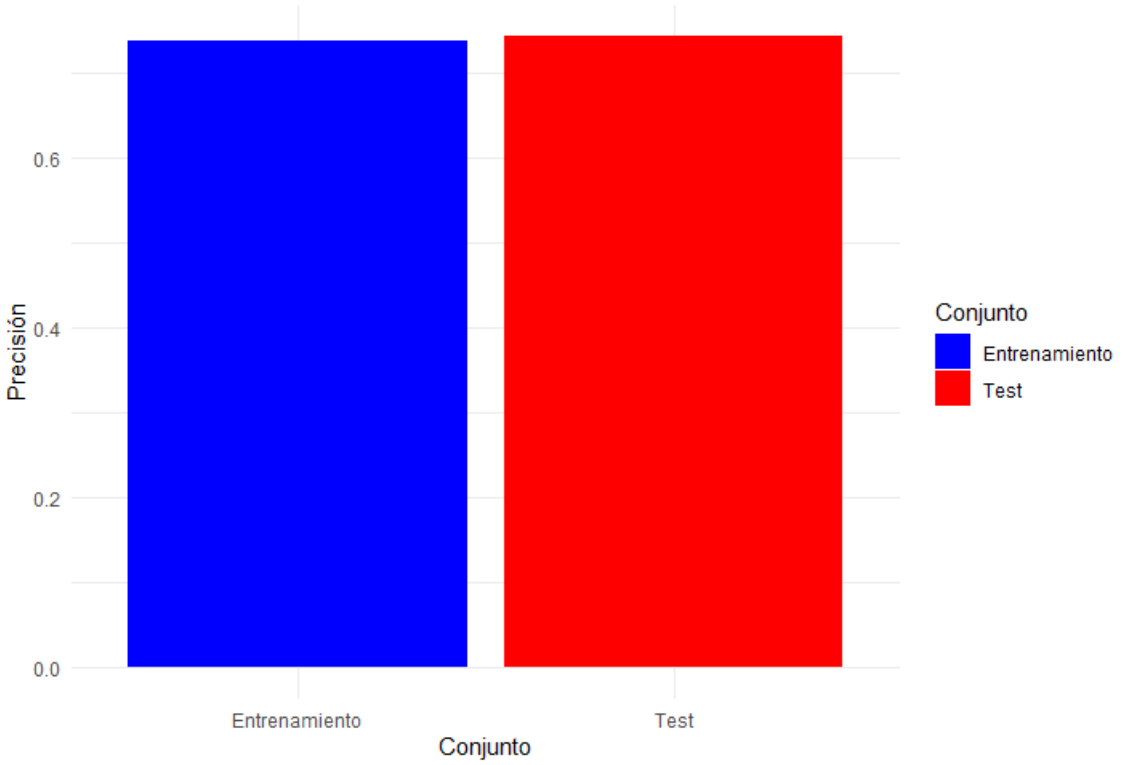

Figura 23: Precisión de k-NN para $k = 11$, en los conjuntos de entrenamiento y test.

3.2. LDA

A continuación se ha implementado el algoritmo LDA para clasificación. De igual manera que para el caso del algoritmo k-NN se ha usado la validación cruzada para medir el rendimiento de este modelo. Sin embargo, primero es necesario comprobar una serie de asunciones para la aplicación de este modelo.

La primera asunción corresponde a que las distribuciones de las distintas características siguen una normal dentro de cada clase (para ello se han procesado los datos de manera conveniente al igual que en el apartado anterior). Cualitativamente se puede ver esto a través de gráficos de densidad. En la figura 24 se muestra esto mismo.

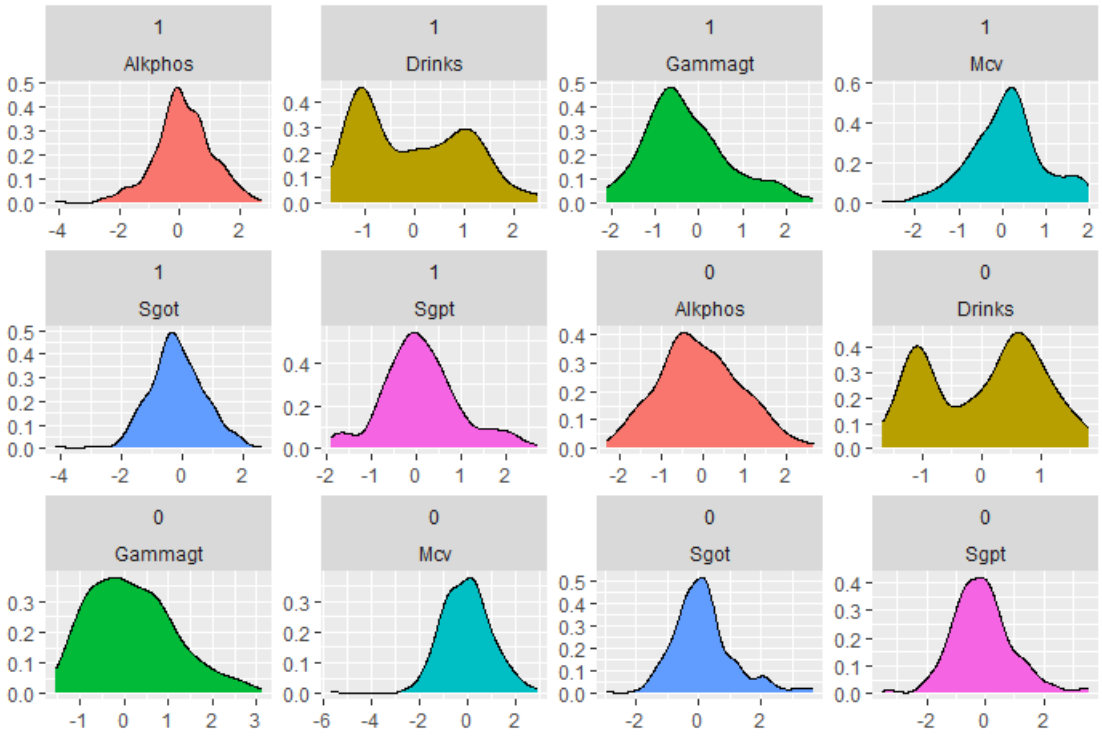

Figura 24: Distribuciones de densidad para cada variable y clase tras procesar los datos.

Podemos intuir que la variable **Drinks** no se distribuye normalmente, pero las demás variables sí que parecen ajustarse a una normal en mayor o menor medida. Cuantitativamente podemos medir esto usando el test de *Shapiro-Wilk's* que también se ha usado en el EDA de este dataset. En este caso nos fijaremos en el estadístico W que está representado para cada variable en la tabla 15. Cuanto más cercano sea el valor de W a 1, mayor semejanza es probable que muestre la distribución de esa variable a una normal.

Clase	Mcv	Alkphos	Sgpt	Sgot	Gammagt	Drinks
1	0.978	0.976	0.975	0.977	0.960	0.881
0	0.966	0.991	0.968	0.949	0.968	0.906

Tabla 15: Estadístico W del test de *Shapiro-Wilk's* para cada variable y clase.

Estos estadísticos junto a la figura 24 y los p-valores que se han obtenido en la tabla 8 nos proporcionan suficiente información acerca de la normalidad de las variables tras ser transformadas. La variable **Alkphos** puede considerarse que se distribuye normalmente, otras variables como **Mcv**, **Sgpt**, **Sgot** y **Gammagt** se aproximan algo a este tipo de distribución y por último, es evidente **Drinks** no se distribuye de manera normal.

Estudiada la asunción de normalidad, resta estudiar la asunción de homogeneidad en las matrices

tanto de varianza como de covarianza.

En la figura 25 se muestra un gráfico de *box-plots* para cada variable diferenciados por clase.

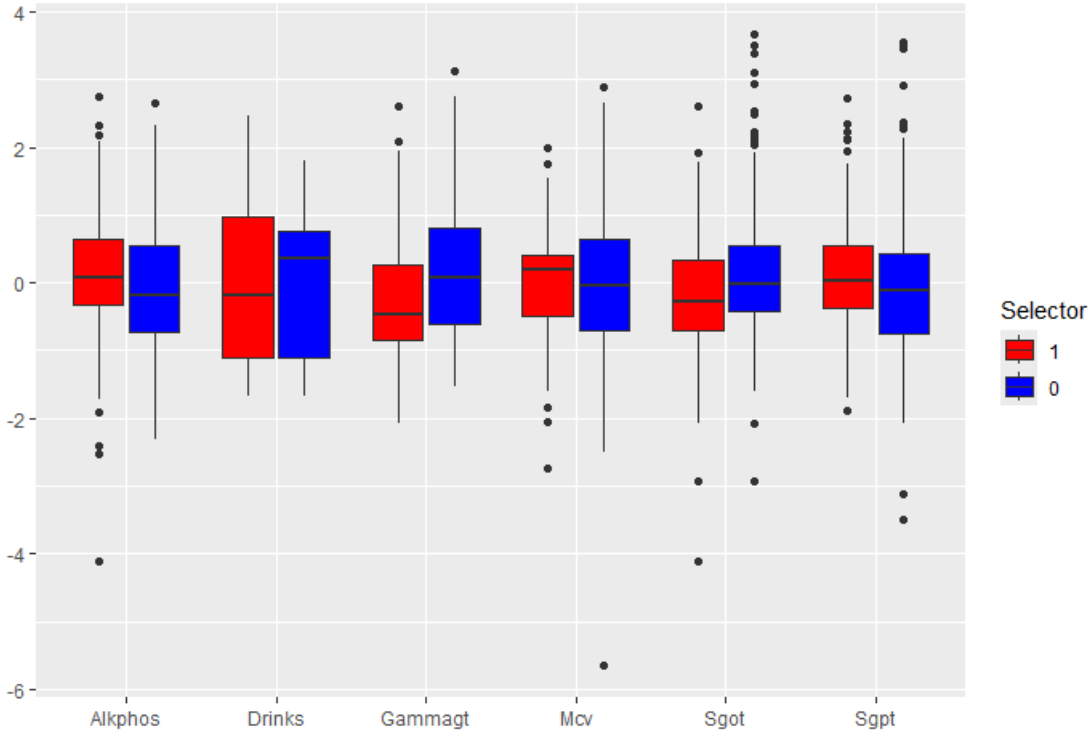

Figura 25: *Box-plots* de las variables transformados, diferenciados por clase.

A la luz del gráfico presentado, la homogeneidad entre las varianzas de las distintas variables no queda muy clara. A continuación, en la tabla 16 se muestran los p-valores resultantes de aplicar el test estadístico de *Bartlett*. La hipótesis nula H_0 de este test consiste en asumir que las varianzas de los grupos son iguales.

Mcv	Alkphos	Sgpt	Sgot	Gammagt	Drinks
0.0059	0.6141	0.0068	0.3807	0.8546	0.1202

Tabla 16: p-valores del test de *Bartlett* para cada variable y clase.

Solo para las variables **Mcv** y **Sgot** se puede rechazar la hipótesis nula, por lo que la varianza parece ser homogénea para la mayoría de variables.

Por otro lado, también se ha realizado el test de *Box's M* para comprobar el resultado de la homogeneidad de las matrices covariantes. La hipótesis nula en este caso es análoga a la del test de *Bartlett* pero para la covarianza entre variables. El p-valor de este test es 0.004, que es menor a

0.05, por lo tanto debemos rechazar la hipótesis nula y en consecuencia, se asume que las matrices de covarianza no son homógeneas entre si.

Teniendo en cuenta el estudio realizado de las asunciones que deben hacerse para la aplicación del algoritmo LDA, los resultados del rendimiento de este algoritmo quedan reflejados en la tabla 17 y la figura 26.

Conjunto	Rendimiento
Test	0.571
Entrenamiento	0.752

Tabla 17: Rendimiento para el modelo LDA.

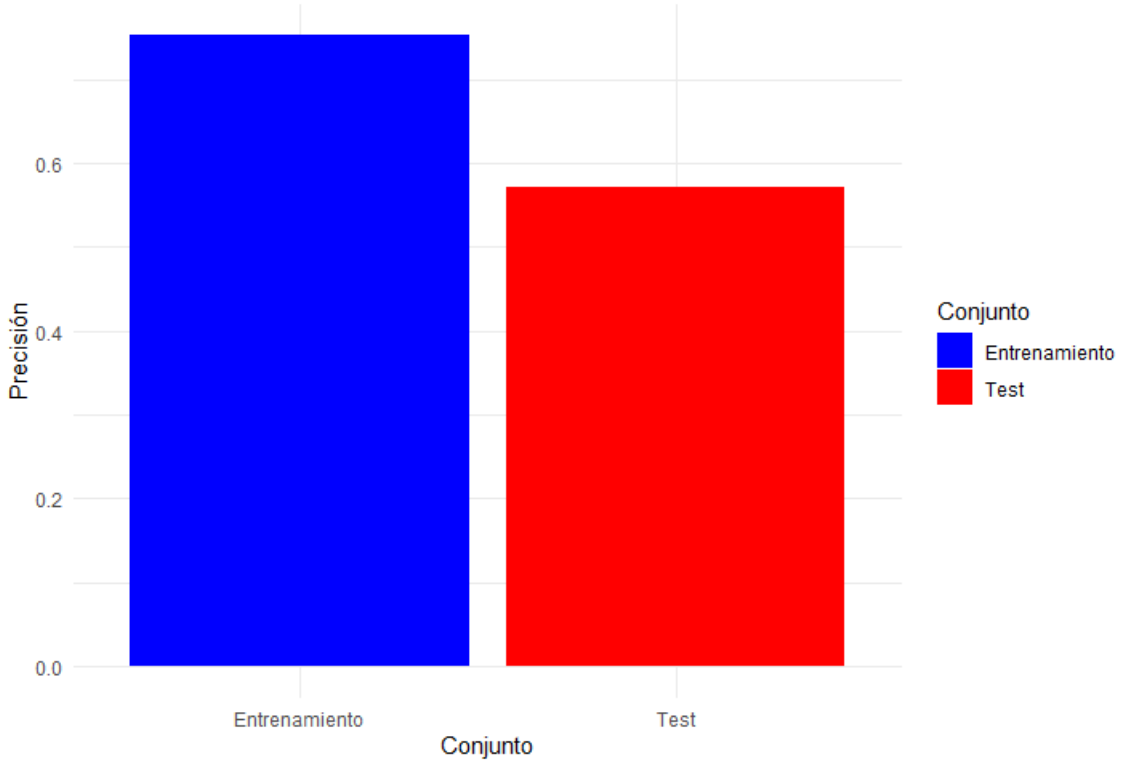

Figura 26: Gráfico que muestra el rendimiento del algoritmo LDA para los conjuntos de entrenamiento y test.

3.3. QDA

En este apartado se aplica el último algoritmo que se va a estudiar en este trabajo, el algoritmo QDA. Este algoritmo es semejante al LDA, pero con la diferencia de que no se asume homogeneidad

en las matrices de varianza y covarianza, ya que permite que las clases se puedan separar mediante curvas cuadráticas a diferencia del algoritmo LDA, el cual separa las clases usando fronteras lineales.

La asunción de normalidad ha sido estudiada en el anterior apartado, por lo cual se muestran directamente los resultados obtenidos al aplicar este algoritmo.

Conjunto	Rendimiento
Test	0.657
Entrenamiento	0.787

Tabla 18: Rendimiento para el modelo QDA.

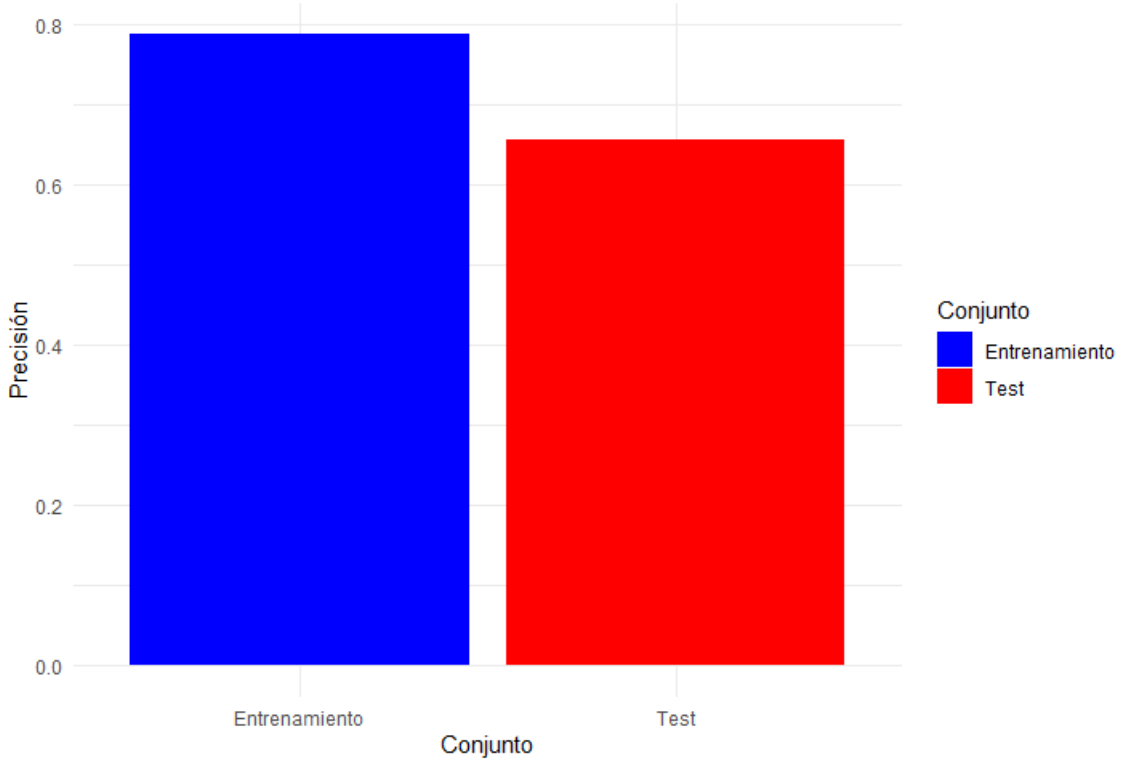

Figura 27: Gráfico que muestra el rendimiento del algoritmo QDA para los conjuntos de entrenamiento y test.

3.4. Comparación de los algoritmos

Finalmente se va a proceder a comparar los resultados que han brindado los 3 algoritmos por separado y se explicarán las posibles razones de las diferencias que surjan entre ellos.

En la figura 28 se muestra un gráfico de barras que contrapone los resultados de los 3 algoritmos.

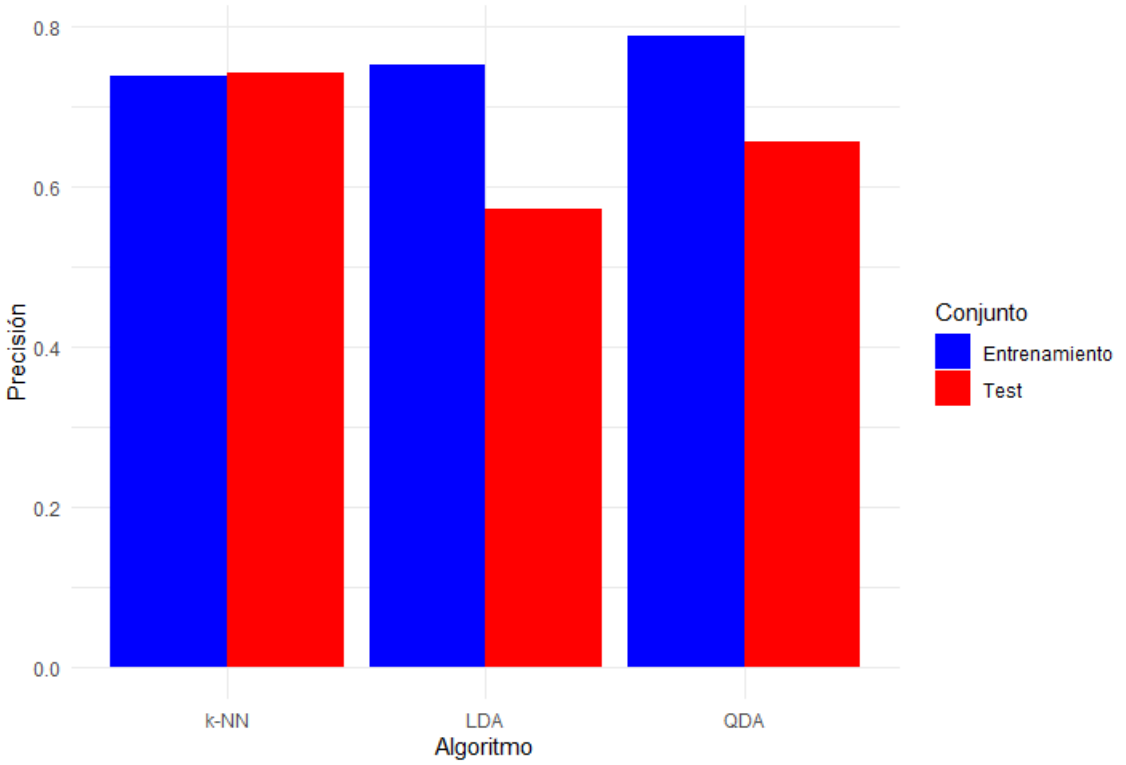

Figura 28: Gráfico que compara el rendimiento de los algoritmos k-NN, LDA y QDA para los conjuntos de entrenamiento y test.

Como se aprecia en el gráfico, es evidente que el algoritmo k-NN es el que mejor resultados proporciona en este caso. Esto se puede deber a que las clases de este dataset no se pueden separar de manera óptima estableciendo una frontera entre ellas, esto es, se encuentran dispersas en el espacio de las dimensiones de las características. Además, es un modelo que no necesita realizar asunciones y por lo tanto, más flexible.

Por otro lado, existe también una diferencia notable entre el algoritmo LDA y QDA, proporcionando un mejor rendimiento este último. Esto, como ya se ha explicado superficialmente en el anterior apartado se debe a que el algoritmo LDA asume que todas las clases comparten la misma matriz de varianza-covarianza. Esta suposición implica que las fronteras de decisión entre las clases deben ser lineales para que proporcione buenos resultados. Sin embargo el algoritmo QDA no asume que las clases compartan la misma matriz de varianza-covarianza. En su lugar, permite que cada clase tenga su propia matriz de covarianza, lo que resulta en fronteras de decisión cuadráticas, no lineales. En este sentido, QDA es más flexible que LDA porque es capaz de captar relaciones más complejas entre las variables.

En resumen, se podría decir que para el dataset *bupa* los algoritmos de clasificación flexibles son los que mejore resultados proporcionan.

A. Código del Trabajo sobre Análisis de Datos

A.1. *california*

```
    #------------------------------------------------------------------------------

#Importamos el dataset
calif <- read.csv("california.dat", comment.char="@", header = FALSE)

#Asignamos manualmente los nombres a las distintas variables
names(calif) <- c("Longitude", "Latitude", "HousingMedianAge",
                  "TotalRooms", "TotalBedrooms", "Population", "Households",
                  "MedianIncome", "MedianHouseValue")
#Vemos las primeras filas del dataset para ver si se ha importado correctamente
head(calif)

#Observamos los atributos y el número de instancias y de características del dataset
str(calif)
dim(calif)

#Comprobamos los tipos de datos atómicos
clase_variables <- sapply(calif, class)
clase_variables

#Generamos un resumen estadístico de las distintas variables
#Aprovechando esto, utilizamos la función skim del paquete skimr
#para ver si hay missing values y obtener más información
summary(calif)
skim(calif)

#------------------------------------------------------------------------------

#Outliers

calif %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(y = valor)) +
  geom_boxplot(fill = "deepskyblue", color = "black", alpha = 0.5) +
  labs(
    x = "",
    y = ""
  ) +
  facet_wrap(~ variable, scales = "free")

#------------------------------------------------------------------------------

#Plots univariables
```

```
calif %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(x = valor)) +
  geom_histogram(bins = 30, fill = "deepskyblue", color = "black") +
  labs(
    x = "",
    y = ""
    ) +
  facet_wrap(~ variable, scales = "free")

#-------------------------------------------------------------------------------

#Plots bivariables

ggpairs(calif,
        lower = list(continuous = wrap("points", size = 0.07,
        alpha = 0.2, color = "deepskyblue")),
        diag = list(continuous = wrap("densityDiag", fill = "purple", alpha = 0.5)),
        upper = list(continuous = "blank")) +
    theme(axis.text.x = element_blank(),
          axis.text.y = element_blank(),
          panel.grid = element_blank()) +
  ggtitle("")

#-------------------------------------------------------------------------------

#Plots para verificar hipótesis

calif %>% ggplot(aes(x = MedianIncome, y = MedianHouseValue)) +
  geom_point(alpha = 0.2, color = "deepskyblue") +
  labs(
    x = "Mediana de Ingresos (10k$)",
    y = "Valor mediano de la vivienda ($)"
  )

calif %>%
  ggplot(aes(x = Longitude, y = Latitude, color = MedianHouseValue)) +
  geom_point(alpha = 0.6, size = 2.5) +
  scale_color_gradient(low = "deepskyblue", high = "purple") +
  labs(
    x = "Longitud (º)",
    y = "Latitud (º)",
    color = "Valor mediano de la vivienda ($)"
  )

calif %>% ggplot(aes(x = TotalRooms, y = TotalBedrooms)) +
  geom_point(alpha = 0.5, color = "deepskyblue") +
```

```
  labs(
    x = "Nº de habitaciones",
    y = "Nº de dormitorios"
  )

calif %>% ggplot(aes(x = Households, y = Population)) +
  geom_point(alpha = 0.5, color = "deepskyblue") +
  labs(
    x = "Nº de hogares",
    y = "Nº de habitantes"
  )

calif %>% ggplot(aes(x = MedianIncome)) +
  geom_density(fill = "deepskyblue", alpha = 0.5) +
  labs(
    x = "Ingreso mediano (10k$)",
    y = "Frecuencia"
  )

#------------------------------------------------------------------------------

#Correlación

y <- calif %>%
  cor()

corrplot(y,, type ="lower", method = "ellipse")

#Reducción de la dimensionalidad

calif_reduced <- calif %>%
  mutate(DistanciaCentro = (Latitude - mean(Latitude))^2 +
  (Longitude - mean(Longitude))^2,
         RoomsPorPopulation = TotalRooms / Population,
         BedroomsPorPopulation = TotalBedrooms / Population,
         HouseholdPorPopulation = Households / Population) %>%
  dplyr::select(-Latitude, -Longitude, -TotalRooms,
  -TotalBedrooms, -Households, -Population)

#------------------------------------------------------------------------------

#Normalidad

calif_reduced %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(sample = valor)) +
  stat_qq() +
```

```
  stat_qq_line()+
  labs(
    x = "",
    y = ""
  ) +
  facet_wrap(~ variable, scales = "free")

shapiro_test_sample <- function(x) {
  x <- sample(x, 5000)
  shapiro.test(x)$p.value
}

shapiro_results <- calif_reduced %>%
  summarise_all(shapiro_test_sample)

print(shapiro_results)

#-------------------------------------------------------------------------------

#Transformaciones

param <- preProcess(calif_reduced, method = c("center", "scale", "YeoJohnson"))
param

calif_yeo_transformed <- predict(param, newdata = calif_reduced)

calif_yeo_transformed %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(x = valor)) +
  geom_histogram(bins = 30, fill = "deepskyblue", color = "black") +
  labs(
    title = "Histogramas de las nuevas variables",
    subtitle = "Transformación: Yeo-Johnson"
  ) +
  facet_wrap(~ variable, scales = "free")

calif_log <- calif_reduced %>%
  mutate_all(log) %>%
  mutate_all(scale)

calif_log %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(x = valor)) +
  geom_histogram(bins = 30, fill = "deepskyblue", color = "black") +
  labs(
    title = "Histogramas de las nuevas variables",
    subtitle = "Transformación: Logarítmica"
  ) +
```

```
  facet_wrap(~ variable, scales = "free")
#-------------------------------------------------------------------------------

#Normalidad de las transformaciones (qqplot y shapiro)

calif_yeo_transformed %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(sample = valor)) +
  stat_qq() +
  stat_qq_line()+
  labs(
    title = "QQ-Plots para las nuevas variables",
  ) +
  facet_wrap(~ variable, scales = "free")

calif_log %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(sample = valor)) +
  stat_qq() +
  stat_qq_line()+
  labs(
    title = "QQ-Plots para las nuevas variables",
  ) +
  facet_wrap(~ variable, scales = "free")


# Verificar la normalidad de las nuevas variables transformadas

shapiro_yeo <- calif_yeo_transformed %>%
  summarise_all(shapiro_test_sample)

shapiro_log <- calif_log %>%
  summarise_all(shapiro_test_sample)

print(rbind(shapiro_yeo, shapiro_log))
```

A.2. *bupa*

```
    #-------------------------------------------------------------------------------
#Importamos el dataset
bupa <- read.csv("bupa.dat", comment.char="@", header = FALSE)

#Asignamos manualmente los nombres a las distintas variables
names(bupa) <- c("Mcv", "Alkphos", "Sgpt",
                 "Sgot", "Gammagt", "Drinks", "Selector")
#Vemos las primeras filas del dataset para ver si se ha importado correctamente
head(bupa)
```

```
#Observamos los atributos y el número de instancias y de características del dataset
str(bupa)
dim(bupa)

#Comprobamos los tipos de datos atómicos
clase_variables <- sapply(bupa, class)
clase_variables

#Podemos pasar a enteras varias variables de este dataset, además hace falta pasar
#a categórica la variable dependiente
bupa_proc <- bupa
bupa_proc[,1:5] <- sapply(bupa_proc[,1:5], as.integer)
bupa_proc$Selector <- factor(bupa_proc$Selector, levels = c(1, 2), labels = c(1, 0))
bupa_proc

#Generamos un resumen estadístico de las distintas variables
#Aprovechando esto, utilizamos la función skim del paquete skimr
#para ver si hay missing values y obtener más información
summary(bupa_proc)
skim(bupa_proc)
#------------------------------------------------------------------------------

#Outliers

bupa %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(y = valor)) +
  geom_boxplot(fill = "deepskyblue", color = "black", alpha = 0.5) +
  labs(
    x = "",
    y = ""
  ) +
  facet_wrap(~ variable, scales = "free")

#------------------------------------------------------------------------------

#Plots univariables

bupa %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(x = valor, fill = Selector)) +
  geom_histogram(bins = 30, fill = "deepskyblue", color = "black") +
  labs(
    x = "",
    y = ""
  ) +
  facet_wrap(~ variable, scales = "free")
```

```
#-------------------------------------------------------------------------------

#Plots bivariables

ggpairs(bupa_proc, mapping = aes(color = Selector),
        lower = list(continuous = wrap("points", size = 0.07, alpha = 0.2)),
        diag = list(continuous = wrap("barDiag")),
        upper = list(continuous = "blank")) +
  scale_fill_manual(values = c("0" = "blue", "1" = "red")) +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        panel.grid = element_blank()) +
  ggtitle("")

#-------------------------------------------------------------------------------
#Plots para verificar hipótesis

#H1
bupa_proc %>% ggplot(aes(x = Drinks)) +
  geom_bar(aes(fill = Selector)) +
  scale_fill_manual(values = c("0" = "blue", "1" = "red")) +
  labs(
    x = "Consumo de alcohol diario en unidades de medias pintas",
    y = "Frecuencia"
  )

#H2
ggpairs(bupa_proc[, 1:5],
        lower = list(continuous = wrap("points", size = 0.07, alpha = 0.2)),
        diag = list(continuous = wrap("barDiag")),
        upper = list(continuous = "blank")) +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        panel.grid = element_blank()) +
  ggtitle("")

bupa_proc %>% ggplot(aes(x = Sgpt, y = Sgot)) +
  geom_point(color = "deepskyblue") +
  labs(
    x = "ALT por Litro (Sgpt/L)",
    y = "AST por Litro (Sgot/L)"
  )

#H3
bupa_bio <- bupa_proc[,1:5]
```

```
bupa_bio %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(x = valor, fill = Selector)) +
  geom_density(fill = "deepskyblue", color = "black") +
  labs(
    x = "",
    y = ""
  ) +
  facet_wrap(~ variable, scales = "free")
#-------------------------------------------------------------------------------
#Correlación

y <- bupa %>%
  cor()

corrplot(y,, type ="lower", method = "ellipse")

#Reducción de dimensionalidad

#Recuento de categoría Selector cuando AST/ALT es menor y mayor a 1

AST_ALT_menor1 <- bupa_proc %>%
  mutate(AST_ALT = (Sgot/Sgpt)) %>% group_by(Selector) %>% filter(AST_ALT<1) %>%
  summarise(N_menor = n())

AST_ALT_mayor1 <- bupa_proc %>%
  mutate(AST_ALT = (Sgot/Sgpt)) %>% group_by(Selector) %>% filter(AST_ALT>1) %>%
  summarise(N_mayor = n())

AST_ALT_menor1
AST_ALT_mayor1


#Creación nueva variable categórica AST_ALT
bupa_new <- bupa_proc %>%
  mutate(AST_ALT = factor(ifelse(Sgot/Sgpt < 1, 1, 0))) %>%
  dplyr::select(-Sgpt, -Sgot)

#Modificación variable Drinks
bupa_new2 <- bupa_new %>%
  mutate(Drinks = factor(ifelse(Drinks >= 15, 1, 0)))
#-------------------------------------------------------------------------------

#Normalidad

bupa[,1:6] %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(sample = valor)) +
```

```
  stat_qq() +
  stat_qq_line()+
  labs(
    x = "",
    y = ""
  ) +
  facet_wrap(~ variable, scales = "free")

shapiro_test <- function(x) {
  shapiro.test(x)$p.value
}

shapiro_results <- apply(bupa[,1:6], 2, shapiro_test)

print(shapiro_results)

#-------------------------------------------------------------------------------

#Transformaciones

bupa_log_transformed <- bupa_proc %>%
  mutate_at(c("Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), log1p) %>%
  mutate_at(c("Mcv", "Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), scale)

bupa_log_transformed[,1:6] %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(x = valor)) +
  geom_histogram(bins = 30, fill = "deepskyblue", color = "black") +
  labs(
    x = "",
    y = ""
  ) +
  facet_wrap(~ variable, scales = "free")

#-------------------------------------------------------------------------------

#Normalidad de las transformaciones (qqplot y shapiro)

bupa_log_transformed[,1:6] %>% gather(key = "variable", value = "valor") %>%
  ggplot(aes(sample = valor)) +
  stat_qq() +
  stat_qq_line()+
  labs(
    title = "QQ-Plots para las nuevas variables",
  ) +
  facet_wrap(~ variable, scales = "free")
```

```
# Verificar la normalidad de las nuevas variables transformadas

shapiro_log <- bupa_log_transformed[,1:6] %>%
  summarise_all(shapiro_test)

shapiro_log
```

B. Código del Trabajo sobre Regresión

```
    #-------------------------------------------------------------------------------
#R-1: Selección de las 5 variables más importantes, Para seleccionar las cinco
#variables más importantes, podríamos basarnos en un análisis de correlación con
#la variable dependiente (MedianHouseValue) y elegir aquellas variables que muestren
#una mayor correlación.Se utiliza el conjunto de datos completo.

# Calcular la correlación con MedianHouseValue
correlaciones <- cor(calif)
correlaciones_importantes <- sort(abs(correlaciones["MedianHouseValue", ]), decreasing = TRUE)
correlaciones_importantes

# Seleccionar las 5 variables más importantes basadas en la correlación, dejamos fuera
#la propia variable independiente que será la primera
top5_variables <- names(correlaciones_importantes)[2:6]
top5_variables

#Aplicamos regresión lineal simple a cada una de las variables seleccionadas,
#medimos las calidad del ajuste con el coeficiente R^2 y el RMSE
modelos <- sapply(top5_variables, function(var) {
  fit <- lm(paste("MedianHouseValue ~", var), data = calif)
  R2 <- summary(fit)$r.squared
  RMSE <- sqrt(sum(fit$residuals^2))/length(fit$residuals)
  c(var = var, R2 = R2, RMSE = RMSE)
})

modelos_df <- data.frame(t(modelos), row.names = NULL, stringsAsFactors = FALSE)

#Realizamos de nuevo los ajuste para graficarlos
fitR1 <- lm(MedianHouseValue ~ MedianIncome, data = calif)
fitR2 <- lm(MedianHouseValue ~ Latitude, data = calif)
fitR3 <- lm(MedianHouseValue ~ TotalRooms, data = calif)
fitR4 <- lm(MedianHouseValue ~ HousingMedianAge, data = calif)
fitR5 <- lm(MedianHouseValue ~ Households, data = calif)

#Dataframe largo para graficar facetando por variable
calif_long <- calif %>%
```

```
  gather(key = "variable", value = "valor",
  MedianIncome, Latitude, TotalRooms, HousingMedianAge, Households)

# Creamos un dataframe con los coeficientes de los modelos
coef_df <- data.frame(
  variable = c("MedianIncome", "Latitude", "TotalRooms", "HousingMedianAge", "Households"),
  ordenada = c(coef(fitR1)[1], coef(fitR2)[1], coef(fitR3)[1], coef(fitR4)[1], coef(fitR5)[1]),
  pendiente = c(coef(fitR1)[2], coef(fitR2)[2], coef(fitR3)[2], coef(fitR4)[2], coef(fitR5)[2])
)

# Unir el dataframe largo con los coeficientes
calif_long <- merge(calif_long, coef_df, by = "variable")

# Graficar usando ggplot2 y facet_wrap
ggplot(calif_long, aes(x = valor, y = MedianHouseValue)) +
  geom_point(alpha = 0.5) +
  geom_abline(aes(intercept = ordenada, slope = pendiente, color = variable),
  show.legend = FALSE) +
  facet_wrap(~variable, scales = "free_x") +
  labs(x = "",
       y = "") +
  theme_minimal()

#Mejor Modelo

ggplot(calif, aes(x = MedianIncome, y = MedianHouseValue)) +
  geom_point(alpha = 0.5) +
  geom_abline(intercept = coef(fitR1)[1], slope = coef(fitR1)[2], color = "red") +
  labs(x = "Ingreso Mediano",
       y = "Valor Mediano de la Casa")

#-------------------------------------------------------------------------------
#R-2: Usar regresión lineal múltiple. Justificar si aporta mejoras respecto al
#modelo lineal simple. Tener en cuenta no linealidad e interacciones.

fitR2_1 <- lm(MedianHouseValue ~ ., data = calif)
summary(fitR2_1)

#Se obtiene un R^2 de 0.6371

# Asignamos nuevos nombres de variables para simplificar
calif_new <- calif
names(calif_new) <- c(paste("X", 1:8, sep = ""), "Y")

#Comenzamos backward selection con bastantes términos y vamos eliminando
#poco a poco.
```

```
fitR2_2 <- lm(Y ~ I(X3^2) + I(X4^2) + I(X5^2) + I(X6^2) + I(X7^2) + I(X8^2) +
                X1*X2 + X1*X3 + X1*X6 + X1*X7 + X1*X8 +
                X2*X3 + X2*X5 + X2*X6 + X2*X8 +
                X3*X4 + X3*X6 + X3*X7 +
                X4*X5 + X4*X8 +
                X5*X6 + X5*X7 +
                X6*X7 + X6*X8 +
                X7*X8,
              data=calif_new)

summary(fitR2_2)

#Eliminamos termino lineal X1 (p-valor 0.762592)

fitR2_3 <- lm(Y ~ I(X3^2) + I(X4^2) + I(X5^2) + I(X6^2) + I(X7^2) + I(X8^2) +
                X1:X2 + X1:X3 + X1:X6 + X1:X7 + X1:X8 +
                X2*X3 + X2*X5 + X2*X6 + X2*X8 +
                X3*X4 + X3*X6 + X3*X7 +
                X4*X5 + X4*X8 +
                X5*X6 + X5*X7 +
                X6*X7 + X6*X8 +
                X7*X8,
              data=calif_new)

summary(fitR2_3)

#Eliminamos término interacción entre X3 y X4 (p-valor 0.66173)

fitR2_4 <- lm(Y ~ I(X3^2) + I(X4^2) + I(X5^2) + I(X6^2) + I(X7^2) + I(X8^2) +
                X1:X2 + X1:X3 + X1:X6 + X1:X7 + X1:X8 +
                X2*X3 + X2*X5 + X2*X6 + X2*X8 +
                X3*X6 + X3*X7 +
                X4*X5 + X4*X8 +
                X5*X6 + X5*X7 +
                X6*X7 + X6*X8 +
                X7*X8,
              data=calif_new)

summary(fitR2_4)

#Eliminamos termino lineal de X7 (p-valor 0.46477)

fitR2_5 <- lm(Y ~ I(X3^2) + I(X4^2) + I(X5^2) + I(X6^2) + I(X7^2) + I(X8^2) +
                X1:X2 + X1:X3 + X1:X6 + X1:X7 + X1:X8 +
                X2*X3 + X2*X5 + X2*X6 + X2*X8 +
```

```
                X3*X6 + X3:X7 +
                X4*X5 + X4*X8 +
                X5*X6 + X5:X7 +
                X6:X7 + X6*X8 +
                X7:X8,
              data=calif_new)

summary(fitR2_5)

#Eliminamos término de interacción entre X1 y X6 (p-valor 0.45293)

fitR2_6 <- lm(Y ~ I(X3^2) + I(X4^2) + I(X5^2) + I(X6^2) + I(X7^2) + I(X8^2) +
                X1:X2 + X1:X3 + X1:X7 + X1:X8 +
                X2*X3 + X2*X5 + X2*X6 + X2*X8 +
                X3*X6 + X3:X7 +
                X4*X5 + X4*X8 +
                X5*X6 + X5:X7 +
                X6:X7 + X6*X8 +
                X7:X8,
              data=calif_new)

summary(fitR2_6)

#Eliminamos término interacción entre X5 y X7 (p-valor 0.09387)

fitR2_7 <- lm(Y ~ I(X3^2) + I(X4^2) + I(X5^2) + I(X6^2) + I(X7^2) + I(X8^2) +
                X1:X2 + X1:X3 + X1:X7 + X1:X8 +
                X2*X3 + X2*X5 + X2*X6 + X2*X8 +
                X3*X6 + X3:X7 +
                X4*X5 + X4*X8 +
                X5*X6 +
                X6:X7 + X6*X8 +
                X7:X8,
              data=calif_new)

summary(fitR2_7)

#Eliminamos término de interacción entre X7 y X8 (p-valor 0.000837)

fitR2_8 <- lm(Y ~ I(X3^2) + I(X4^2) + I(X5^2) + I(X6^2) + I(X7^2) + I(X8^2) +
                X1:X2 + X1:X3 + X1:X7 + X1:X8 +
                X2*X3 + X2*X5 + X2*X6 + X2*X8 +
                X3*X6 + X3:X7 +
                X4*X5 + X4*X8 +
                X5*X6 +
                X6:X7 + X6*X8,
```

```
              data=calif_new)

summary(fitR2_8)

#Eliminamos el término cuadrático de X4 (p-valor 0.000692)

fitR2_9 <- lm(Y ~ I(X3^2) + I(X5^2) + I(X6^2) + I(X7^2) + I(X8^2) +
                X1:X2 + X1:X3 + X1:X7 + X1:X8 +
                X2*X3 + X2*X5 + X2*X6 + X2*X8 +
                X3*X6 + X3:X7 +
                X4*X5 + X4*X8 +
                X5*X6 +
                X6:X7 + X6*X8,
              data=calif_new)

summary(fitR2_9)

#Quitamos el término de interacción entre X5 y X6 (p-valor 1.88e-06)

fitR2_10 <- lm(Y ~ I(X3^2) + I(X5^2) + I(X6^2) + I(X7^2) + I(X8^2) +
                 X1:X2 + X1:X3 + X1:X7 + X1:X8 +
                 X2*X3 + X2*X5 + X2*X6 + X2*X8 +
                 X3*X6 + X3:X7 +
                 X4*X5 + X4*X8 +
                 X6:X7 + X6*X8,
               data=calif_new)

summary(fitR2_10)

#Quitamos término cuadrático de X3 (p-valor 5.24e-09)

fitR2_11 <- lm(Y ~ I(X5^2) + I(X6^2) + I(X7^2) + I(X8^2) +
                 X1:X2 + X1:X3 + X1:X7 + X1:X8 +
                 X2*X3 + X2*X5 + X2*X6 + X2*X8 +
                 X3*X6 + X3:X7 +
                 X4*X5 + X4*X8 +
                 X6:X7 + X6*X8,
               data=calif_new)

summary(fitR2_11)

#Quitamos término interacción entre X6 y X7 (p-valor 1.28e-12)

fitR2_12 <- lm(Y ~ I(X5^2) + I(X6^2) + I(X7^2) + I(X8^2) +
                 X1:X2 + X1:X3 + X1:X7 + X1:X8 +
                 X2*X3 + X2*X5 + X2*X6 + X2*X8 +
```

```
                  X3*X6 + X3:X7 +
                  X4*X5 + X4*X8 +
                  X6*X8,
                data=calif_new)

summary(fitR2_12)

#Quitamos término cuadrático de X7 (p-valor 7.08e-09)

fitR2_13 <- lm(Y ~ I(X5^2) + I(X6^2) + I(X8^2) +
                  X1:X2 + X1:X3 + X1:X7 + X1:X8 +
                  X2*X3 + X2*X5 + X2*X6 + X2*X8 +
                  X3*X6 + X3:X7 +
                  X4*X5 + X4*X8 +
                  X6*X8,
                data=calif_new)

summary(fitR2_13)

#Quitamos término de interacción entre X1 y X7 (p-valor 2.26e-08)

fitR2_14 <- lm(Y ~ I(X5^2) + I(X6^2) + I(X8^2) +
                  X1:X2 + X1:X3 + X1:X8 +
                  X2*X3 + X2*X5 + X2*X6 + X2*X8 +
                  X3*X6 + X3:X7 +
                  X4*X5 + X4*X8 +
                  X6*X8,
                data=calif_new)

summary(fitR2_14)

#Quitamos término de interacción entre X2 y X6 (p-valor 6.48e-11)

fitR2_15 <- lm(Y ~ I(X5^2) + I(X6^2) + I(X8^2) +
                  X1:X2 + X1:X3 + X1:X8 +
                  X2*X3 + X2*X5 + X2*X8 +
                  X3*X6 + X3:X7 +
                  X4*X5 + X4*X8 +
                  X6*X8,
                data=calif_new)

summary(fitR2_15)

#Eliminamos término lineal de X6 (p-valor 0.0770)

fitR2_16 <- lm(Y ~ I(X5^2) + I(X6^2) + I(X8^2) +
```

```
                  X1:X2 + X1:X3 + X1:X8 +
                  X2*X3 + X2*X5 + X2*X8 +
                  X3:X6 + X3:X7 +
                  X4*X5 + X4*X8 +
                  X6:X8,
                data=calif_new)

summary(fitR2_16)

#Eliminamos término interacción entre X2 y X5 (p-valor 0.0179)

fitR2_17 <- lm(Y ~ I(X5^2) + I(X6^2) + I(X8^2) +
                  X1:X2 + X1:X3 + X1:X8 +
                  X2*X3 + X2*X8 +
                  X3:X6 + X3:X7 +
                  X4*X5 + X4*X8 +
                  X6:X8,
                data=calif_new)

summary(fitR2_17)

#Todos los términos del ajuste tienen un p-valor<2e-16

#Por lo tanto el mejor modelo de regresión lineal múltiple encontrado para términos
#cuadráticos en términos de interpretabilidad y rendimiento según el threshold escogido
#es el siguiente

fitR2_final <- lm(Y ~ I(X5^2) + I(X6^2) + I(X8^2) +
                    X1:X2 + X1:X3 + X1:X8 +
                    X2*X3 + X2*X8 +
                    X3:X6 + X3:X7 +
                    X4*X5 + X4*X8 +
                    X6:X8,
                  data=calif_new)


R2_final <- summary(fitR2_final)$r.squared
RMSE_final <- sqrt(sum(fitR2_final$residuals^2))/length(fitR2_final$residuals)

R2_multi_lin <- summary(fitR2_1)$r.squared
RMSE_multi_lin <- sqrt(sum(fitR2_1$residuals^2))/length(fitR2_1$residuals)

resultados_R2 <- data.frame("Regresion" = c("Lineal sobre todas las variables",
                                            "No-linealidad e interacciones"),
                            "R2" = c(R2_multi_lin, R2_final),
                            "RMSE" = c(RMSE_multi_lin, RMSE_final))
```

```
#Se ha conseguido incrementar el valor de R^2 en 0.05 y se ha reducido un poco el RSME.

#------------------------------------------------------------------------------

#R-3: Aplicar el algoritmo k-NN para regresión.

library(kknn)

#k-NN
nombre <- "california"
run_knn_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-5-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@", header=FALSE)
  file <- paste(x, "-5-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@", header=FALSE)
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") {
    test <- x_tra
  }
  else { test <- x_tst
  }
  fitMulti <- kknn(Y~.,x_tra,test)
  yprime <- fitMulti$fitted.values
  sum(abs(test$Y-yprime)^2)/length(yprime)
}

knnMSEtrain<-mean(sapply(1:5,run_knn_fold,nombre,"train"))
knnMSEtest<-mean(sapply(1:5,run_knn_fold,nombre,"test"))


#No lineal
run_lm_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-5-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@" , header=FALSE )
  file <- paste(x, "-5-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@" , header=FALSE )
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tst)[In+1] <- "Y"
```

```
  if (tt == "train") { test <- x_tra }
  else { test <- x_tst }
  fitMulti <- lm(Y ~ I(X5^2) + I(X6^2) + I(X8^2) +
                X1:X2 + X1:X3 + X1:X8 +
                X2*X3 + X2*X8 +
                X3:X6 + X3:X7 +
                X4*X5 + X4*X8 +
                X6:X8,
              data=x_tra)
  yprime <- predict(fitMulti, test)
  sum(abs(test$Y-yprime)^2)/length(yprime)
}

lmMSEtrain<-mean(sapply(1:5,run_lm_fold, nombre, "train"))
lmMSEtest<-mean(sapply(1:5,run_lm_fold, nombre, "test"))

#-------------------------------------------------------------------------------

#R-4: Comparación de algoritmos.


run_lm2_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-5-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@" , header=FALSE )
  file <- paste(x, "-5-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@" , header=FALSE )
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") { test <- x_tra }
  else { test <- x_tst }
  fitMulti <- lm(Y ~ .,
                 data=x_tra)
  yprime <- predict(fitMulti, test)
  sum(abs(test$Y-yprime)^2)/length(yprime) ##MSE
}

lm2MSEtrain<-mean(sapply(1:5,run_lm2_fold, nombre, "train"))
lm2MSEtest<-mean(sapply(1:5,run_lm2_fold, nombre, "test"))

#Leemos la tabla con los errores medios de test tras haber cambiado el MSE del
#dataset california

resultados <- read.csv("regr_test_alumnos.csv")
```

```
tablatst <- cbind(resultados[,2:dim(resultados)[2]])
colnames(tablatst) <- names(resultados)[2:dim(resultados)[2]]
rownames(tablatst) <- resultados[,1]

#Leemos la tabla con los errores medios de entrenamiento tras cambiar el MSE del
#dataset california
resultados <- read.csv("regr_train_alumnos.csv")
tablatra <- cbind(resultados[,2:dim(resultados)[2]])
colnames(tablatra) <- names(resultados)[2:dim(resultados)[2]]
rownames(tablatra) <- resultados[,1]

#Wilcoxon
difs_tst <- (tablatst[,1] - tablatst[,2]) / tablatst[,1]
difs_tra <- (tablatra[,1] - tablatra[,2]) / tablatra[,1]
wilc_1_2_tst <- cbind(ifelse(difs_tst<0, abs(difs_tst)+0.1, 0+0.1),
ifelse (difs_tst>0, abs(difs_tst)+0.1, 0+0.1))
wilc_1_2_tra <- cbind(ifelse(difs_tra<0, abs(difs_tra)+0.1, 0+0.1),
ifelse (difs_tra>0, abs(difs_tra)+0.1, 0+0.1))

colnames(wilc_1_2_tst) <- c(colnames(tablatst)[1], colnames(tablatst)[2])
colnames(wilc_1_2_tra) <- c(colnames(tablatst)[1], colnames(tablatst)[2])

LMvsKNNtst <- wilcox.test(wilc_1_2_tst[,1], wilc_1_2_tst[,2],
alternative = "two.sided", paired=TRUE)
Rmas_tst <- LMvsKNNtst$statistic
pvalue_tst <- LMvsKNNtst$p.value
LMvsKNNtst <- wilcox.test(wilc_1_2_tst[,2], wilc_1_2_tst[,1],
alternative = "two.sided", paired=TRUE)
Rmenos_tst <- LMvsKNNtst$statistic

LMvsKNNtra <- wilcox.test(wilc_1_2_tra[,1], wilc_1_2_tra[,2],
alternative = "two.sided", paired=TRUE)
Rmas_tra <- LMvsKNNtra$statistic
pvalue_tra <- LMvsKNNtra$p.value
LMvsKNNtra <- wilcox.test(wilc_1_2_tra[,2], wilc_1_2_tra[,1],
alternative = "two.sided", paired=TRUE)
Rmenos_tra <- LMvsKNNtra$statistic

#Friedman
test_friedman_tst <- friedman.test(as.matrix(tablatst))
test_friedman_tst

test_friedman_tra <- friedman.test(as.matrix(tablatra))
test_friedman_tra

#Holm
```

```
tam <- dim(tablatst)
groups <- rep(1:tam[2], each=tam[1])
pairwise.wilcox.test(as.matrix(tablatst), groups, p.adjust = "holm", paired = TRUE)
pairwise.wilcox.test(as.matrix(tablatra), groups, p.adjust = "holm", paired = TRUE)
```

C. Código del Trabajo sobre Clasificación

```
    #------------------------------------------------------------------------------
#C-1: Utilizar el algoritmo k-NN probando con diferentes valores de k.
#Elegir el que considere más adecuado para su conjunto de datos.
#Analice qué ocurre en los valores de precisión en training y test con los
#diferentes valores de k.

library(class)
library(MASS)
library(biotools)

#función para aplicar k-NN a los 10-folds de datos
knn_fold <- function(i, x, k_value) {
  file <- paste(x, "-10-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@" , header=FALSE )
  file <- paste(x, "-10-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@" , header=FALSE )
  names(x_tra) <- c("Mcv", "Alkphos", "Sgpt",
                    "Sgot", "Gammagt", "Drinks", "Selector")
  names(x_tst) <- c("Mcv", "Alkphos", "Sgpt",
                    "Sgot", "Gammagt", "Drinks", "Selector")
  #Preprocesamiento
  x_tra_log_transformed <- x_tra %>%
    mutate(Selector = factor(Selector, levels = c(1, 2), labels = c(1, 0))) %>%
    mutate_at(c("Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), log1p) %>%
    mutate_at(c("Mcv", "Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), scale)

  x_tst_log_transformed <- x_tst %>%
    mutate(Selector = factor(Selector, levels = c(1, 2), labels = c(1, 0))) %>%
    mutate_at(c("Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), log1p) %>%
    mutate_at(c("Mcv", "Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), scale)

  #Predicción
  knn.pred.tst <- knn(x_tra_log_transformed[,-7], x_tst_log_transformed[,-7],
  x_tra_log_transformed$Selector, k = k_value)
  knn.pred.tra <- knn(x_tra_log_transformed[,-7], x_tra_log_transformed[,-7],
  x_tra_log_transformed$Selector, k = k_value)

  #Medidas de calidad
  t_tst <- table(knn.pred.tst, x_tst_log_transformed$Selector)
```

```
  val_acc_rate_tst <- sum(diag(t_tst)) / nrow(x_tst_log_transformed)
  t_tra <- table(knn.pred.tra, x_tra_log_transformed$Selector)
  val_acc_rate_tra <- sum(diag(t_tra)) / nrow(x_tra_log_transformed)

  c(val_acc_rate_tst, val_acc_rate_tra)
}

#Valores de k a probar

k_valores <- 1:20

#Aplicar función a cada valor de k distinto
nombre <- "bupa"
resultados <- mapply(function(k) {
  r_tst <- mean(sapply(1:10, knn_fold, nombre, k)[1])
  r_tra <- mean(sapply(1:10, knn_fold, nombre, k)[2])
  c(r_tst, r_tra)
}, k_valores)

#Pasar vector de resultados a dataframe
resultados_knn_df <- data.frame(
  k = k_valores,
  acc_test = resultados[1, ],
  acc_train = resultados[2, ]
)

#Gráfico para visualizar rendimiento
ggplot(resultados_knn_df, aes(x = k)) +
  geom_line(aes(y = acc_test, color = "Precisión Test")) +
  geom_line(aes(y = acc_train, color = "Precisión Entrenamiento")) +
  labs(x = "Valor de k", y = "Precisión") +
  scale_color_manual("", values = c("Precisión Test" = "red", "Precisión Entrenamiento" = "blue"))

#Para el conjunto de datos, el valor de k que proporciona mejor rendimiento sería 11 o 12.
knn_fold_best <- function(i, x) {
  file <- paste(x, "-10-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@" , header=FALSE )
  file <- paste(x, "-10-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@" , header=FALSE )
  names(x_tra) <- c("Mcv", "Alkphos", "Sgpt",
                    "Sgot", "Gammagt", "Drinks", "Selector")
  names(x_tst) <- c("Mcv", "Alkphos", "Sgpt",
                    "Sgot", "Gammagt", "Drinks", "Selector")
  #Preprocesamiento
  x_tra_log_transformed <- x_tra %>%
    mutate(Selector = factor(Selector, levels = c(1, 2), labels = c(1, 0))) %>%
```

```
    mutate_at(c("Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), log1p) %>%
    mutate_at(c("Mcv", "Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), scale)

  x_tst_log_transformed <- x_tst %>%
    mutate(Selector = factor(Selector, levels = c(1, 2), labels = c(1, 0))) %>%
    mutate_at(c("Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), log1p) %>%
    mutate_at(c("Mcv", "Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), scale)

  #Predicción
  knn.pred.tst <- knn(x_tra_log_transformed[,-7], x_tst_log_transformed[,-7],
  x_tra_log_transformed$Selector, k = 11)
  knn.pred.tra <- knn(x_tra_log_transformed[,-7], x_tra_log_transformed[,-7],
  x_tra_log_transformed$Selector, k = 11)

  #Medidas de calidad
  t_tst <- table(knn.pred.tst, x_tst_log_transformed$Selector)
  val_acc_rate_tst <- sum(diag(t_tst)) / nrow(x_tst_log_transformed)
  t_tra <- table(knn.pred.tra, x_tra_log_transformed$Selector)
  val_acc_rate_tra <- sum(diag(t_tra)) / nrow(x_tra_log_transformed)

  c(val_acc_rate_tst, val_acc_rate_tra)
}

r_knn_tst <- mean(sapply(1:10, knn_fold_best, nombre)[1])
r_knn_tra <- mean(sapply(1:10, knn_fold_best, nombre)[2])

resultados_knn_best_df <- data.frame(
  Conjunto = c("Test", "Entrenamiento"),
  Precision = c(r_knn_tst, r_knn_tra)
)

#Gráfico para visualizar rendimiento
ggplot(resultados_knn_best_df, aes(x = Conjunto, y = Precision, fill = Conjunto)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(x = "Conjunto", y = "Precisión") +
  scale_fill_manual(values = c("Test" = "red", "Entrenamiento" = "blue"))
#-------------------------------------------------------------------------------
#C-2: Utilizar el algoritmo LDA para clasificar. No olvide comprobar las asunciones.

#Comprobamos las asunciones de partida, para ello se mostrarán gráficos y se realizarán
#los tests pertinentes.

#Normalidad en las variables predictoras (tras las transformaciones, log + scale)
#(Ver parte de EDA para mas información)

#Gráfico
```

```
plot_data_C2 <- bupa_log_transformed %>% pivot_longer(-Selector)

ggplot(plot_data_C2) +
  geom_density(aes(x = value, fill = name)) +
  labs(
    x = "",
    y = ""
  ) +
  facet_wrap(~ Selector + name, scales = "free")+
  theme(legend.position = "none")

#Test de Shapiro-Wilk's
my_shapiro <- function(a) {
  shapiro.test(a)$statistic
}

bupa_log_transformed %>%
  group_by(Selector) %>%
  summarise(across(Mcv:Drinks, my_shapiro))

#Homogeneidad de las matrices de varianza y covarianza

#Gráfico
ggplot(plot_data_C2) +
  geom_boxplot(aes(x=name, y=value, fill=Selector)) +
  labs(
    x = "",
    y = ""
  )+
  scale_fill_manual(values = c("1" = "red", "0" = "blue"))

#Test de Bartlett
my_bartlett <- function(variable, clase) {
  bartlett.test(variable ~ clase)$p.value
}

bupa_log_transformed %>%
  summarise(across(Mcv:Drinks, ~ my_bartlett(.x, bupa_log_transformed$Selector)))

#Test boxM

boxM(bupa_log_transformed[, 1:6], bupa_log_transformed$Selector)

#Aplicar LDA
lda_fold <- function(i, x) {
  file <- paste(x, "-10-", i, "tra.dat", sep="")
```

```
  x_tra <- read.csv(file, comment.char="@" , header=FALSE )
  file <- paste(x, "-10-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@" , header=FALSE )
  names(x_tra) <- c("Mcv", "Alkphos", "Sgpt",
                    "Sgot", "Gammagt", "Drinks", "Selector")
  names(x_tst) <- c("Mcv", "Alkphos", "Sgpt",
                    "Sgot", "Gammagt", "Drinks", "Selector")
  #Preprocesamiento
  x_tra_log_transformed <- x_tra %>%
    mutate(Selector = factor(Selector, levels = c(1, 2), labels = c(1, 0))) %>%
    mutate_at(c("Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), log1p) %>%
    mutate_at(c("Mcv", "Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), scale)

  x_tst_log_transformed <- x_tst %>%
    mutate(Selector = factor(Selector, levels = c(1, 2), labels = c(1, 0))) %>%
    mutate_at(c("Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), log1p) %>%
    mutate_at(c("Mcv", "Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), scale)

  #Predicción
  model.lda <- lda(Selector ~ ., data = x_tra_log_transformed)

  lda.pred.tra <- predict(model.lda, x_tra_log_transformed)
  lda.pred.tst <- predict(model.lda, x_tst_log_transformed)

  #Medidas de calidad

  t_tst <- table(lda.pred.tst$class, x_tst_log_transformed$Selector)
  val_acc_rate_tst <- sum(diag(t_tst)) / nrow(x_tst_log_transformed)
  t_tra <- table(lda.pred.tra$class, x_tra_log_transformed$Selector)
  val_acc_rate_tra <- sum(diag(t_tra)) / nrow(x_tra_log_transformed)

  c(val_acc_rate_tst, val_acc_rate_tra)
}

r_lda_tst <- mean(sapply(1:10, lda_fold, nombre)[1])
r_lda_tra <- mean(sapply(1:10, lda_fold, nombre)[2])

#Pasar vector de resultados a dataframe
resultados_lda_df <- data.frame(
  Conjunto = c("Test", "Entrenamiento"),
  Precision = c(r_lda_tst, r_lda_tra)
)

#Gráfico para visualizar rendimiento
ggplot(resultados_lda_df, aes(x = Conjunto, y = Precision, fill = Conjunto)) +
  geom_bar(stat = "identity", position = "dodge") +
```

```
  labs(x = "Conjunto", y = "Precisión") +
  scale_fill_manual(values = c("Test" = "red", "Entrenamiento" = "blue"))
#-------------------------------------------------------------------------------
#C-3: Utilizar el algoritmo QDA para clasificar. No olvide comprobar las asunciones.

#Asuncion de normalidad estudiadas en C2

#Aplicar QDA
qda_fold <- function(i, x) {
  file <- paste(x, "-10-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@" , header=FALSE )
  file <- paste(x, "-10-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@" , header=FALSE )
  names(x_tra) <- c("Mcv", "Alkphos", "Sgpt",
                    "Sgot", "Gammagt", "Drinks", "Selector")
  names(x_tst) <- c("Mcv", "Alkphos", "Sgpt",
                    "Sgot", "Gammagt", "Drinks", "Selector")
  #Preprocesamiento
  x_tra_log_transformed <- x_tra %>%
    mutate(Selector = factor(Selector, levels = c(1, 2), labels = c(1, 0))) %>%
    mutate_at(c("Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), log1p) %>%
    mutate_at(c("Mcv", "Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), scale)

  x_tst_log_transformed <- x_tst %>%
    mutate(Selector = factor(Selector, levels = c(1, 2), labels = c(1, 0))) %>%
    mutate_at(c("Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), log1p) %>%
    mutate_at(c("Mcv", "Alkphos", "Gammagt", "Sgpt", "Sgot", "Drinks"), scale)

  #Predicción
  model.qda <- qda(Selector ~ ., data = x_tra_log_transformed)

  qda.pred.tra <- predict(model.qda, x_tra_log_transformed)
  qda.pred.tst <- predict(model.qda, x_tst_log_transformed)

  #Medidas de calidad

  t_tst <- table(qda.pred.tst$class, x_tst_log_transformed$Selector)
  val_acc_rate_tst <- sum(diag(t_tst)) / nrow(x_tst_log_transformed)
  t_tra <- table(qda.pred.tra$class, x_tra_log_transformed$Selector)
  val_acc_rate_tra <- sum(diag(t_tra)) / nrow(x_tra_log_transformed)

  c(val_acc_rate_tst, val_acc_rate_tra)
}

r_qda_tst <- mean(sapply(1:10, qda_fold, nombre)[1])
r_qda_tra <- mean(sapply(1:10, qda_fold, nombre)[2])
```

```
#Pasar vector de resultados a dataframe
resultados_qda_df <- data.frame(
  Conjunto = c("Test", "Entrenamiento"),
  Precision = c(r_qda_tst, r_qda_tra)
)

#Gráfico para visualizar rendimiento
ggplot(resultados_qda_df, aes(x = Conjunto, y = Precision, fill = Conjunto)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(x = "Conjunto", y = "Precisión") +
  scale_fill_manual(values = c("Test" = "red", "Entrenamiento" = "blue"))
#-------------------------------------------------------------------------------
#C-4: Comparar los resultados de los tres algoritmos.

#Comparamos mediante rendimiento por precisión

resultados_modelos <- rbind(resultados_knn_best_df, resultados_lda_df,
                            resultados_qda_df)

resultados_modelos <- cbind (resultados_modelos, Modelo = rep(c("k-NN", "LDA", "QDA"), each = 2))

ggplot(resultados_modelos, aes(x = Modelo, y = Precision, fill = Conjunto)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(x = "Algoritmo", y = "Precisión") +
  scale_fill_manual(values = c("Test" = "red", "Entrenamiento" = "blue"))
```