Introducción a la Ciencia de Datos 2024-2025

Master Ciencia de Datos

Universidad de Granada

Proyecto Final

Miguel García López

Índice

Índice de figuras

Índice de cuadros

1. Introducción

En el presente proyecto final de la asignatura de **Introducción a la Ciencia de Datos**, se llevarán a cabo tres trabajos fundamentales en el ámbito de la ciencia de datos, centrados en el análisis de datos y la creación de modelos predictivos. Estos trabajos son los siguientes:

- **Análisis exploratorio de datos (EDA):** Análisis de dos conjuntos de datos, uno correspondiente a un problema de **regresión** y otro a un problema de **clasificación**.
- **Regresión:** Aplicación de modelos de regresión lineal simple y la realización de pruebas y análisis correspondientes.
- **Clasificación:** Ajuste de diversos modelos de clasificación, incluyendo *kNN*, y realización de comparaciones entre los diferentes algoritmos.

Para cada uno de los problemas mencionados, se dispondrán de dos conjuntos de datos diferentes: uno para la clasificación y otro para la regresión.

El documento se planteará con la siguiente estructura, primero los **EDAs** correspondientes a regresión y clasificación y después las tareas correspondientes al modelado de los predictores para cada problema junto con sus sub-tareas.

1.1. Tareas

1.1.1. Análisis de Datos

En este apartado, el estudiante deberá llevar a cabo un análisis preliminar de los dos conjuntos de datos asignados, denominados `datasetR` y `datasetC`. Este análisis deberá abarcar los siguientes puntos:

- **A-1 Descripción del tipo de datos de entrada:** Es necesario especificar el formato de los datos (por ejemplo, lista, data frame, etc.), el número de filas y columnas, así como los tipos de datos atómicos presentes.
- **A-2 Cálculo de medidas estadísticas:** Se deben calcular medidas estadísticas como la media, la desviación estándar y otras métricas descriptivas relevantes.
- **A-3 Gráficos:** Se deberán generar gráficos adecuados que faciliten la visualización de los datos.
- **A-4 Descripción del conjunto de datos:** A partir de los análisis anteriores, se debe proporcionar una descripción detallada del conjunto de datos.

1.1.2. Regresión

En este apartado, el estudiante deberá utilizar el conjunto de datos `datasetR` para llevar a cabo las siguientes tareas:

- **R-1 Regresión lineal simple:** Se aplicará el algoritmo de regresión lineal simple a cada regresor (variable de entrada) para obtener los modelos correspondientes. En el caso de que `datasetR` contenga más de cinco regresores, se deberá seleccionar de manera justificada los cinco más relevantes. Posteriormente, se elegirá el modelo que se considere más adecuado según las medidas de calidad conocidas.
- **R-2 Regresión lineal múltiple:** Se aplicará el algoritmo de regresión lineal múltiple. Es necesario justificar si el modelo obtenido representa una mejora respecto al modelo elegido en el paso anterior. En este análisis se deberán tener en cuenta posibles interacciones entre las variables y la no linealidad.
- **R-3 Algoritmo k-NN para regresión:** Se aplicará el algoritmo k-NN para la regresión.
- **R-4 Comparación de resultados:** Se realizará una comparación entre los resultados obtenidos con los dos algoritmos de regresión múltiple. Además, se realizarán comparativas con un tercer modelo, el modelo de regresión M5', cuyos resultados ya están disponibles en las tablas de resultados.

Nota: Al final de las transparencias de las clases de laboratorio sobre regresión, el estudiante encontrará aclaraciones adicionales sobre los apartados R-1 a R-4. Estas aclaraciones serán más comprensibles una vez que se haya realizado el trabajo práctico correspondiente.

1.1.3. Clasificación

En este apartado, el estudiante deberá utilizar el conjunto de datos `datasetC` asignado para realizar las siguientes tareas:

- **C-1 Algoritmo k-NN para clasificación:** Se aplicará el algoritmo k-NN utilizando diferentes valores de k. Se deberá seleccionar el valor de k más adecuado para el conjunto de datos. Además, se analizará el comportamiento de la precisión en los conjuntos de entrenamiento y prueba con los distintos valores de k.
- **C-2 Algoritmo LDA para clasificación:** Se aplicará el algoritmo de Análisis Discriminante Lineal (LDA) para la clasificación. Es fundamental verificar que se cumplen las asunciones necesarias para la correcta aplicación del algoritmo.

- **C-3 Algoritmo QDA para clasificación:** Se aplicará el algoritmo de Análisis Discriminante Cuadrático (QDA) para la clasificación, verificando también las asunciones necesarias.
- **C-4 Comparación de algoritmos:** Finalmente, se realizará una comparación entre los resultados obtenidos con los tres algoritmos mencionados.

2. EDA Regresión

El *dataset* asignado para el problema de regresión es el de **wankara**. Este conjunto de datos incluye una serie de variables relacionadas con el tiempo de la ciudad Turca de Ankara.

2.1. Resumen de los datos

Se obtienen algunos de los estadísticos relacionados con el *dataset*.

Variable	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Max_temperature	23.00	46.40	60.80	61.56	77.40	100.00
Min_temperature	-7.10	26.60	36.00	37.08	48.20	65.50
Dewpoint	-3.10	28.50	36.80	36.29	45.30	57.60
Precipitation	0.0000	0.0000	0.0000	0.0541	0.0000	4.0000
Sea_level_pressure	29.46	29.83	29.96	29.98	30.11	30.60
Standard_pressure	26.30	26.69	26.77	26.78	26.87	27.18
Visibility	0.200	7.400	8.300	7.718	8.600	11.500
Wind_speed	0.000	3.110	5.060	5.393	7.250	18.000
Max_wind_speed	2.19	10.20	12.70	13.32	16.10	57.40
Mean_temperature	7.90	36.70	48.50	49.56	63.30	81.80

Cuadro 1: Summary statistics for weather-related variables.

El conjunto de datos cuenta con 1609 filas (observaciones) y 9 columnas (características). Todas las características del conjunto son de tipo numérico (son números reales) y no hay ningún valor faltante, por lo que no es necesario realizar ningún tipo de imputación sobre los datos o eliminar observaciones (si fuese plausible).

Dado los datos obtenidos en la tabla 1 se puede observar que los datos son bastante simples. No hay gran desviación en la mayoría de variables. La variable **Precipitación** si tiene datos más extremos, ya que desde su mínimo hasta el tercer cuartil sus valores son cero y su máximo cuatro, lo que significa un salto muy grande.

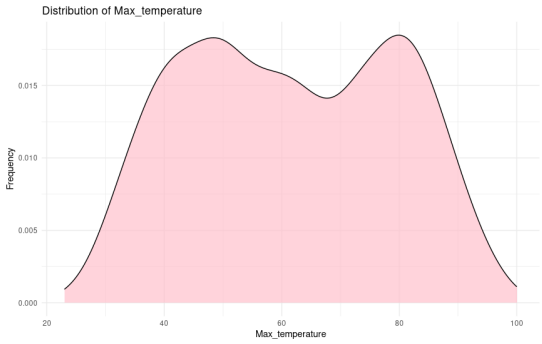

Figura 1: Max temperature.

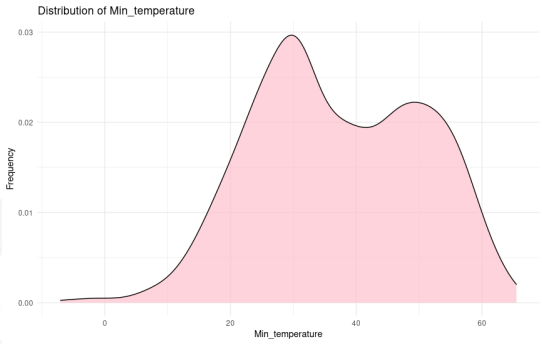

Figura 2: Min temperature.

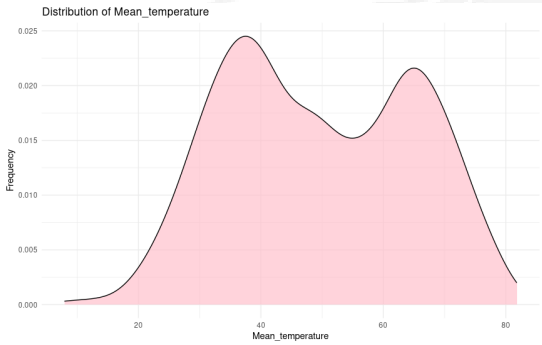

Figura 3: Mean temperature.

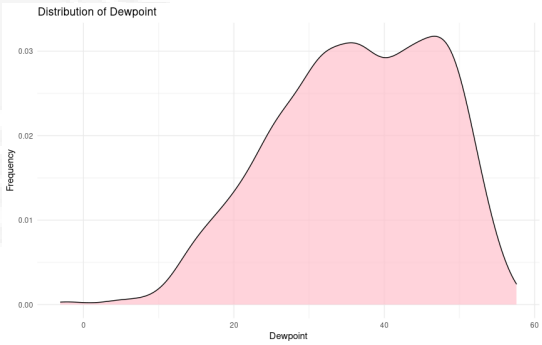

Figura 4: Dewpoint.

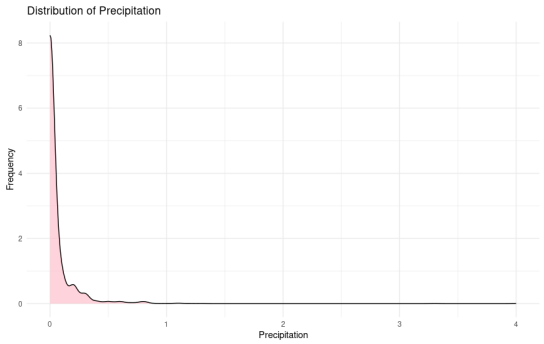

Figura 5: Precipitation.

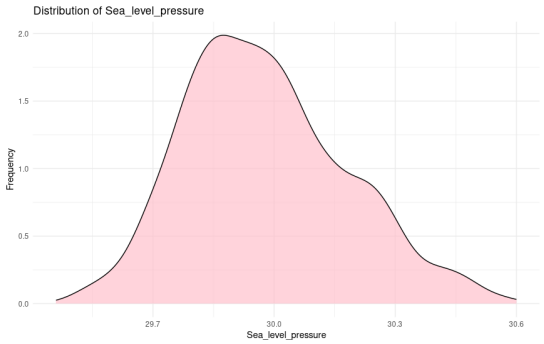

Figura 6: Sea level pressure.

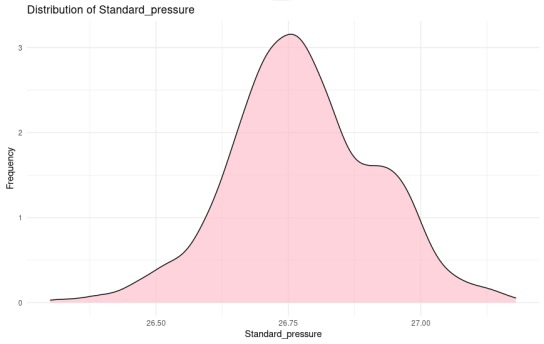

Figura 7: Standard pressure.

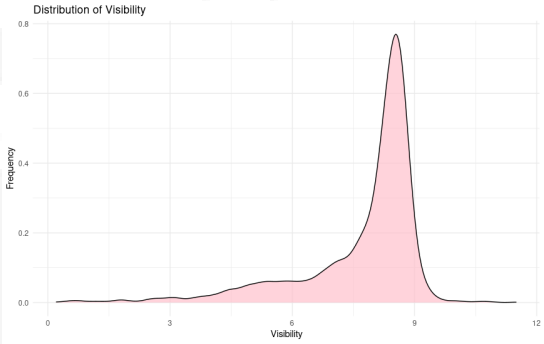

Figura 8: Visibility.

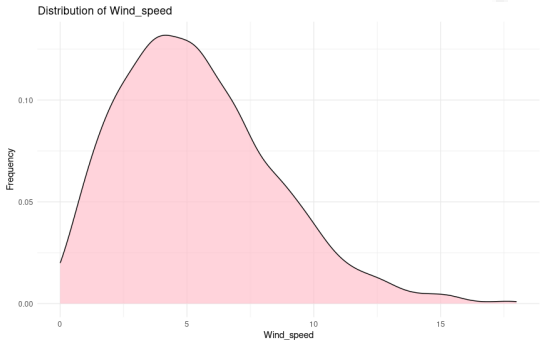

Figura 9: Wind speed.

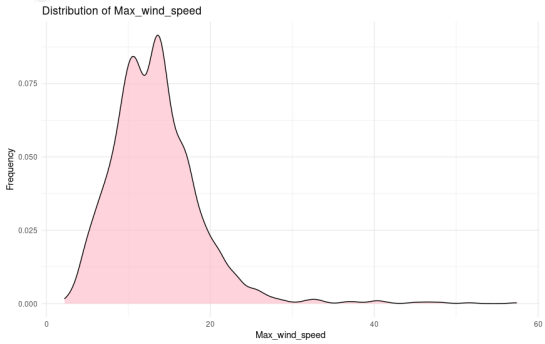

Figura 10: Max wind speed.

2.2. Normalidad

Puede observarse en las figuras que hay algunas variables con dos picos, las relacionadas con la temperatura concretamente (figuras 1 2 3). Esto suele indicar una distribución de tipo bimodal. Hay otras variables, como la precipitación 5, visibilidad 8 y las relativas al viento, que tienen asimetría. Algunas de ellas son muy extremas, como la precipitación. Aquellas un poco más leves y que son asimétricas hacia la derecha pueden "corregirseçon algunas transformaciones, como la logarítmica.

La bimodalidad en las temperaturas puede darse por varios motivos, pero el que parece más probable es la toma de datos en distintas zonas geográficas de Ankara, dando lugar a distintas modas. El calentamiento climático podría ser factible, pero no sería tan evidente y se necesitarían tomas de muchos años.

Se estandarizan los datos para que estos obtengan una media de 1 y varianza 0. Escalar permite que todas las variables contribuyan equitativamente a la distancia calculada, evitando que una variable desproporcionada influya en la detección de *outliers*.

$$z = \frac{x - \mu}{\sigma}$$

Figura 11: standardization formula (Z-score).

Se procede a obtener gráficos de *QQPlot* para todas las variables. Intuitivamente se puede saber que hay algunas que no se van a adecuar a la línea de la normalidad representada por los cuantiles de la distribución normal, pero así es posible descartarlas visualmente.

Como puede observarse en la figura 12, tan solo las relativas a las presiones parecen adecuarse. Es interesante com la precipitación, que tiene una asimetría extrema, tiene la cola derecha muy larga en el *QQPlot*. Las variables de presión al nivel del mar y presión estándar tienen una forma más cercana a la normal, por ello se les aplica un test de *Shapiro* para comprobar su normalidad. El test de *Shapiro* rechaza siempre y cuando haya evidencia estadística de que la distribución no sigue una normal.

Como se observa en la figura 13, se obtienen valores muy significativos, por lo que se rechaza la hipótesis nula y no se pueden considerar las distribuciones de las variables como normales.

2.3. Outliers univariantes

Se procede a analizar las gráficas de *Boxplots* para visualizar posibles valores anómalos en los datos.

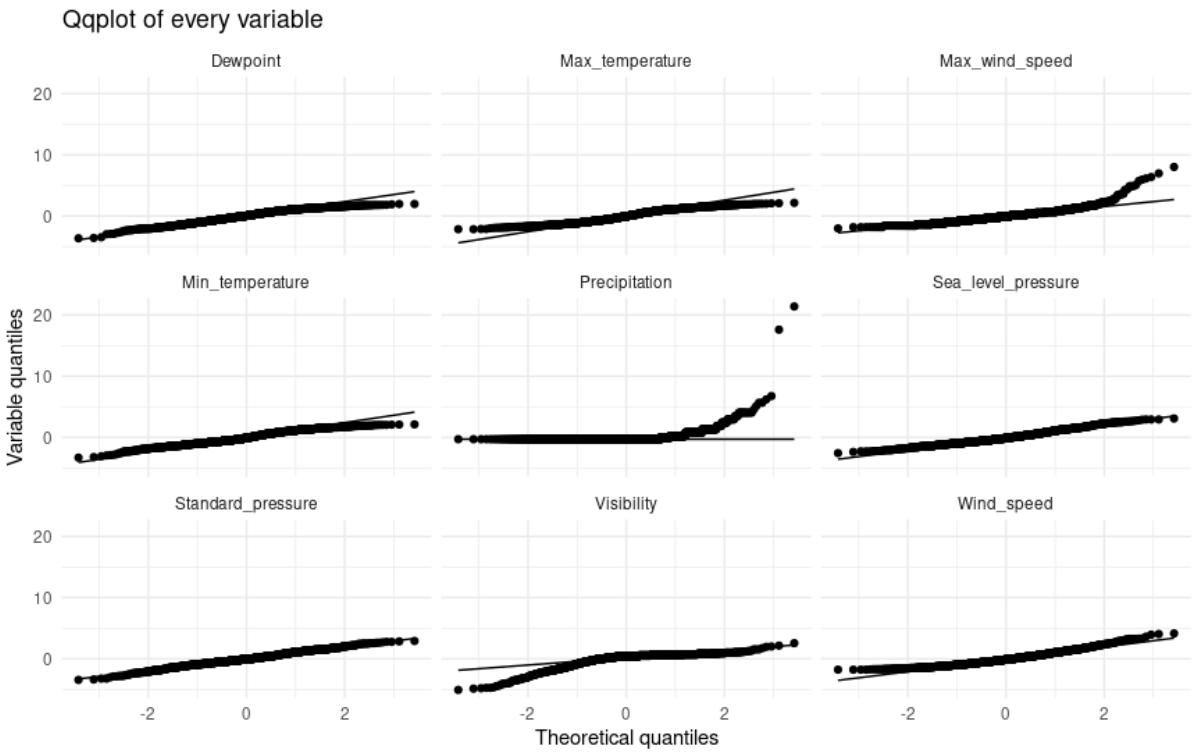

Figura 12: QQplot for every variable in dataset.

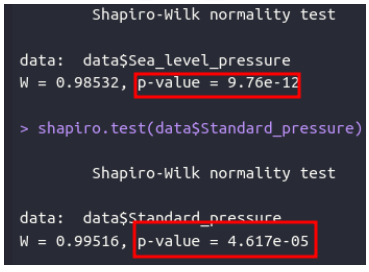

Figura 13: Shapiro test for normality.

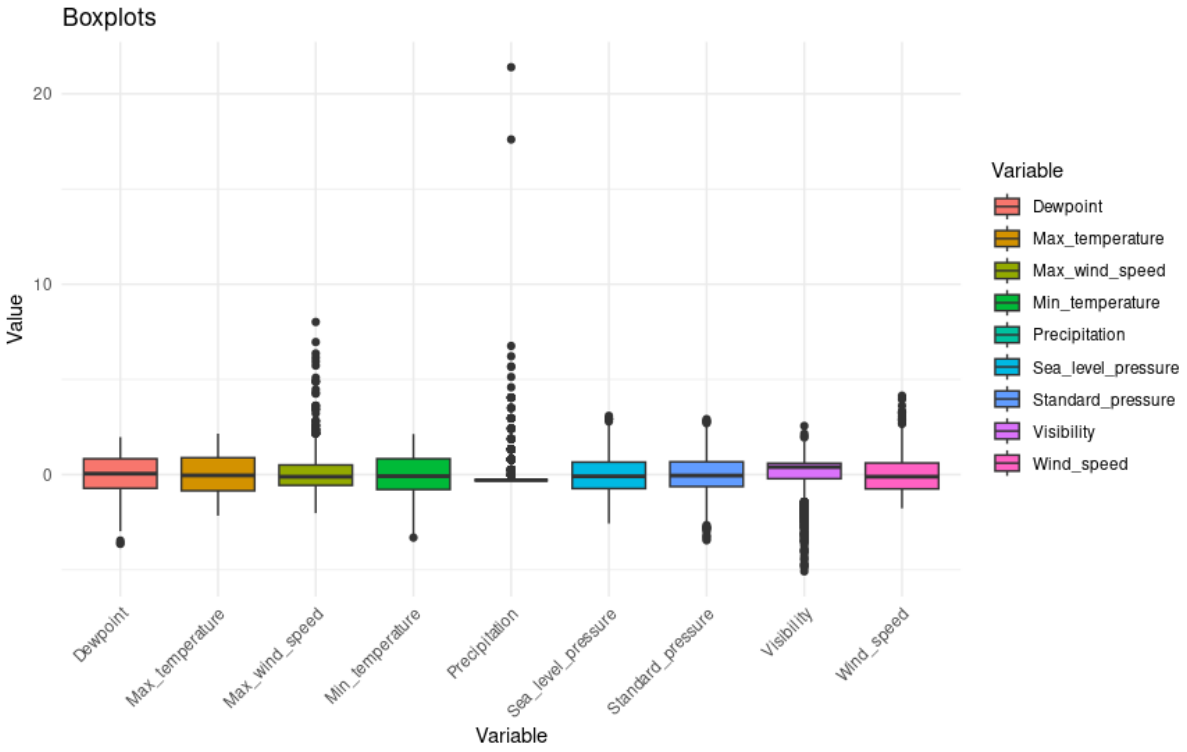

Figura 14: Boxplots for regression.

Puede observarse en la figura 14 como la variable de precipitación tiene lo que podría considerarse valores anómalos extremos. En el resumen obtenido anteriormente de los datos ya fue detectado. Ankara es una zona de pocas precipitaciones, por tanto la mayoría de valores son 0, pero eso no implica que el día que llueva sea un valor irreal. Es anómalo por su rareza, pero debería persistir en el conjunto de datos, pues es un valor real.

Observando el resto de las variables y sus valore máximos y mínimos no se podría considerar que existan *outliers* univariantes, ya que son valores posibles y nada irreales. Con ello se pretende decir que aunque salgan datos anómalos usando métodos como la distancia intercuartil, se decide dejar los datos tal y como están. Además no se cuenta con conocimiento experto para poder asegurar la eliminación de datos.

2.4. Outliers multivariantes

Se procede a analizar la normalidad del conjunto para todas sus dimensiones, esto con el objetivo de encontrar *outliers multivariantes*. Para ello se utiliza primero un *test* como el **HZ** o *Henze-Zirkler*. El test de HZ evalúa si un conjunto de datos sigue una distribución normal multivariada. Es decir, extiende la idea de pruebas de normalidad univariadas (como Shapiro-Wilk) a dimensiones superiores.

Esto se hace para poder utilizar la distancia *Mahalanobis* en la detección de *outliers*.

```
> mvn(data = data, mvnTest = "hz")
$multivariateNormality
           Test      HZ p value MVN
1 Henze-Zirkler 3.87392       0  NO
```

Figura 15: Hz test for multivariate normality.

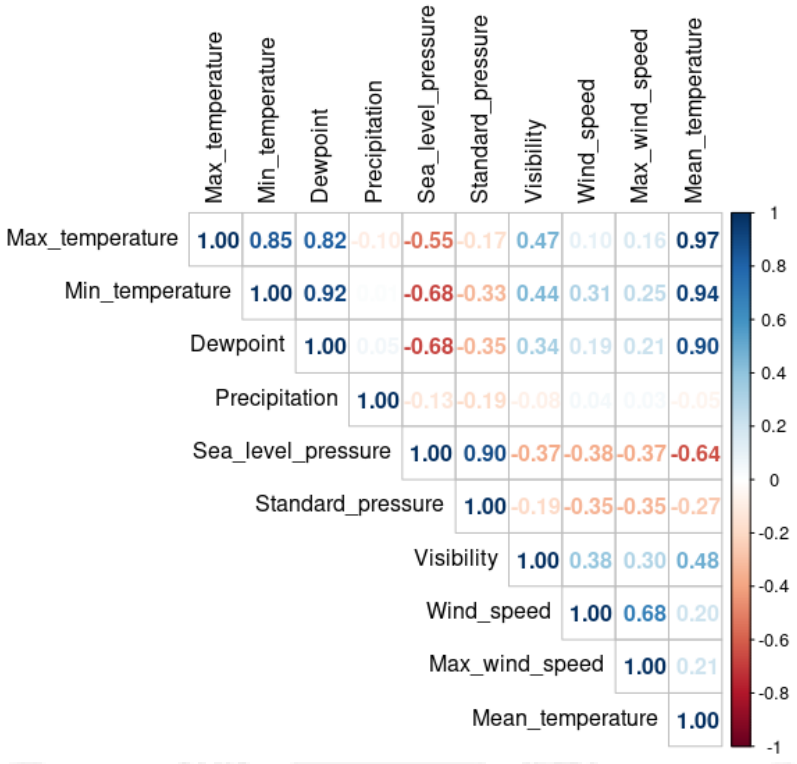

Figura 16: Correlation between variables.

Como indica la figura 15, no puede considerarse que el conjunto sea normal. Esto es obvio, si las variables por separado no lo eran, raramente lo serán en conjunto. Dados estos resultados, no es conveniente usar la distancia de *Mahalanobis*, ya que se asume normalidad multivariante. Se podrían utilizar métodos como **LOF**, pero se exceden el dominio de la asignatura y además no sería coherente con el análisis descrito anteriormente sobre *outliers* univariantes, ya que no sería posible por parte del estudiante saber si los datos son realmente irreales como para eliminarlos.

2.5. Correlación

Se analiza la correlación entre variables del *dataset*.

Según se observa en la figura 16 hay correlación (de tipo lineal) entre algunas variables, sobre todo entre aquellas relativas a la temperatura (mínima, máxima y media). El punto de condensación también está altamente correlacionado con la temperatura media

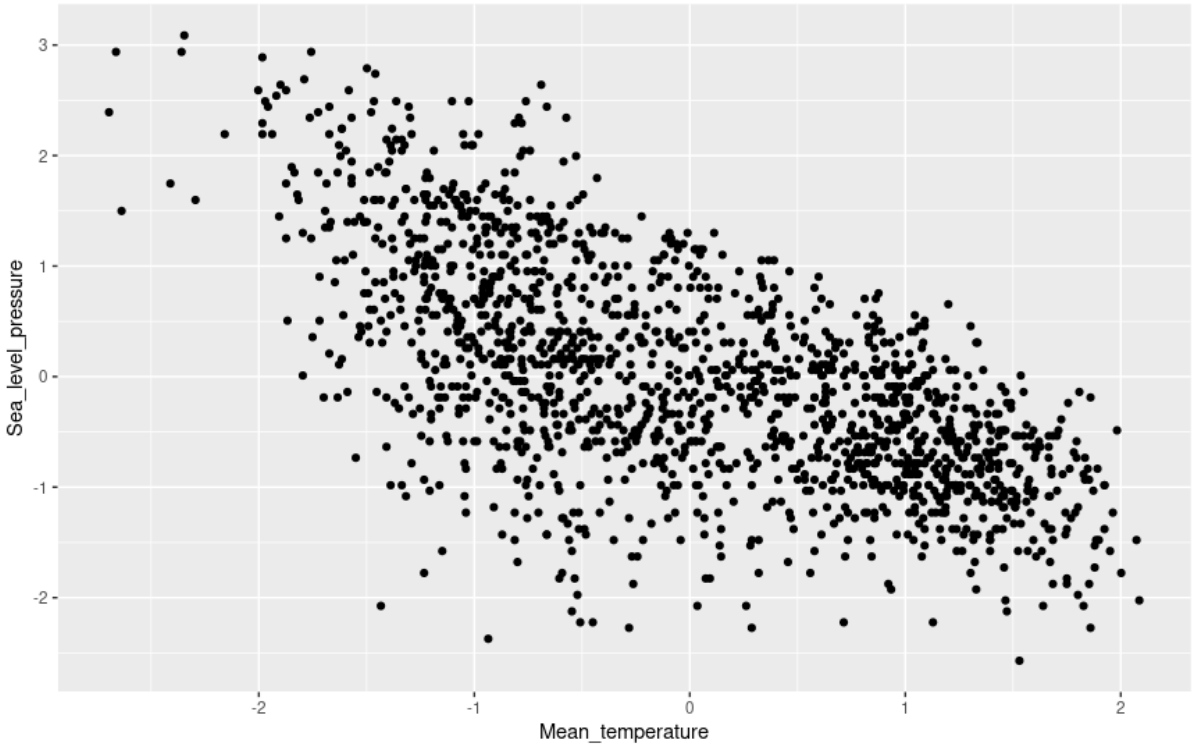

Figura 17: Mean temperature value with sea level pressure.

(variable objetivo). El hecho de que el punto de condensación o *dewpoint* también esté correlacionado con la variable objetivo refuerza la idea de que las condiciones atmosféricas, como la humedad y la temperatura, están relacionadas con el comportamiento de la variable de interés. En particular, el *dewpoint* es un indicador importante de la humedad en el aire, lo que podría influir directamente en el fenómeno que se está modelando.

Las variables de presión a nivel del agua también tienen correlación con las temperaturas. La presión atmosférica disminuye con el aumento de la elevación, por lo que podría intuirse que la temperatura media disminuye en lugares con poca altitud en este conjunto de datos, como puede observarse en la figura 17.

2.5.1. Construcción de características

Variables como *max_temperature* y *min_temperature* parecen redundantes, al igual que las de presión. Si bien correlación no implica que dos variables aporten la misma información, pues pueden ser ambas necesarias, quizá se pueda crear una nueva variable o varias nuevas. Pasar de máximo y mínimo de temperatura a *diff_temperature*. Esta nueva característica sería la suma de las dos anteriormente mencionadas, lo mismo puede realizarse con las relativas a la presión.

Realizando esta simple construcción, se elimina posible multicolinealidad entre las variables de temperatura con *mean_temperature*. Se obtiene una nueva variable que tiene muy

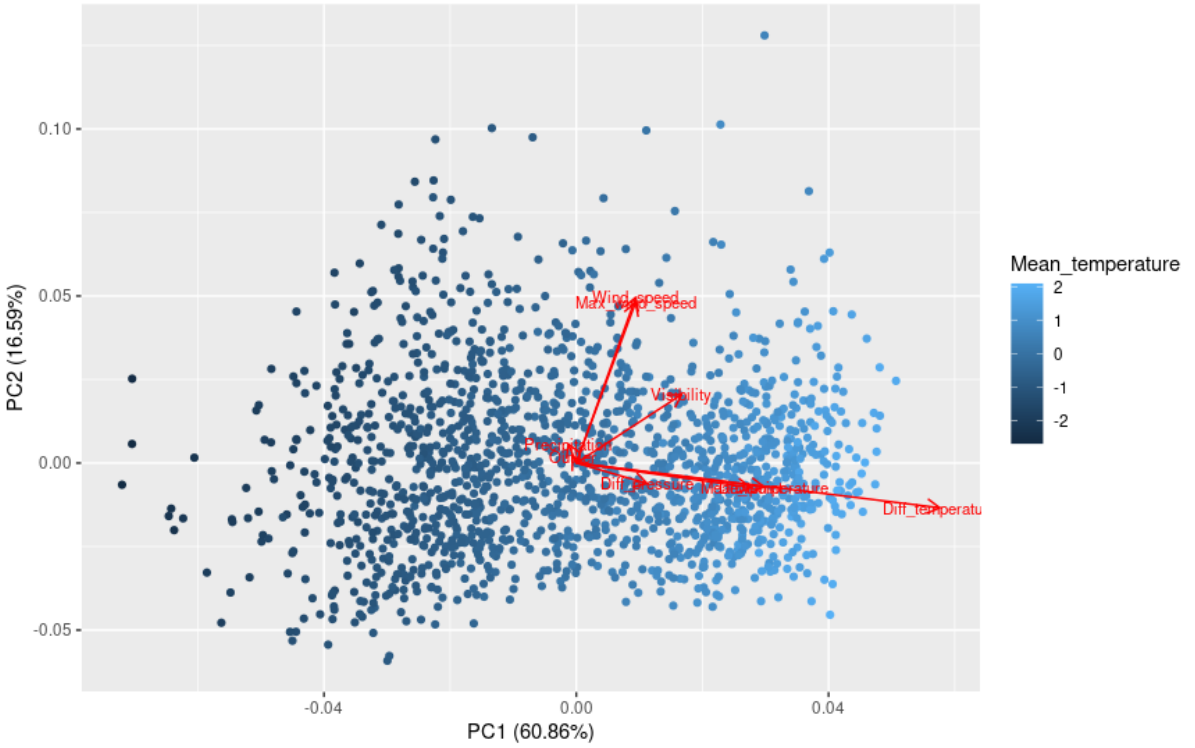

Figura 18: Biplot for Wankara.

buena correlación y no se pierde apenas información.

2.6. PCA

Si bien no es necesario reducir características en este conjunto de datos, pues son solo diez, es interesante para poder analizar ciertos gráficos como el *Biplot*.

Como se puede observar en el *Biplot* 18, allí donde el punto de condensación aumenta, lo hace la temperatura media, así como la presión. La nueva característica de diferencia de temperatura, al ser la resta entre mínimos y máximos de temperatura, crece en dirección perpendicular a la temperatura media. Calculando de nuevo la matriz de correlación, se podría observar que esta nueva característica ya no está correlacionada con la variable objetivo (linealmente al menos).

3. EDA Clasificación

3.1. Resumen de los datos

Se obtiene el siguiente resumen de los datos usando el comando de *summary* de **R**.

Variable	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
T3resin	65.0	103.0	110.0	109.6	117.5	144.0
Thyroxin	0.500	7.100	9.200	9.805	11.300	25.300
Triiodothyronine	0.20	1.35	1.70	2.05	2.20	10.00
Thyroidstimulating	0.10	1.00	1.30	2.88	1.70	56.40
TSH_value	-0.700	0.550	2.000	4.199	4.100	56.300
Class	1.000	1.000	1.000	1.442	2.000	3.000

Cuadro 2: Summary statistics for thyroid-related variables.

Hay solo 5 variables en todo el *dataset* y un total de 215 observaciones. No hay variables faltantes, por lo que no es necesario utilizar ningún tipo de técnica de imputación. En cuanto a las variables, según lo visto en la tabla 2, se puede decir que en general es necesario realizar un escalado de los datos. Variables como *TSH_value* o *Thyroidstimulating* se distribuyen muy ampliamente y con valores en escalas muy diferentes. Además ha de remarcarse que es necesario transformar la variable de clase (etiqueta *Class*) a un factor para poder tratarlas como categorías.

Existen tres clases, $1, 2$ y 3. La proporción de clases en el conjunto de datos es la descrita en la tabla 3.

Class	Total Items	Proportion
1	150	0.6977
2	35	0.1628
3	30	0.1395
Total	**215**	**1.0000**

Cuadro 3: Class distribution and proportions.

Claramente existe un desbalance de clases a favor de la clase 1, las otras dos están proporcionadas entre sí. Es interesante analizar este desbalance más a fondo.

3.2. Normalidad

Como se observa en las figuras 19,20,21,22,23 estas presentan un alto grado de asimetría. Queda claro con un primer vistazo que no cumplen la normalidad, de hecho no pasan los test de *Shapiro*.

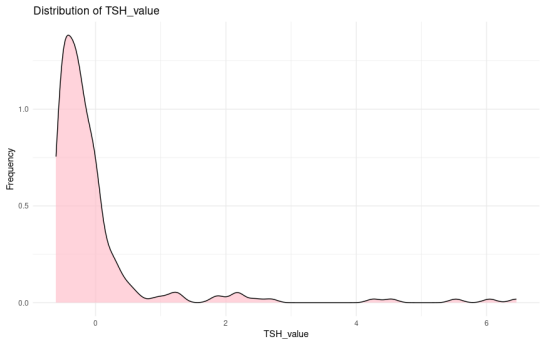

Figura 19: THS_value.

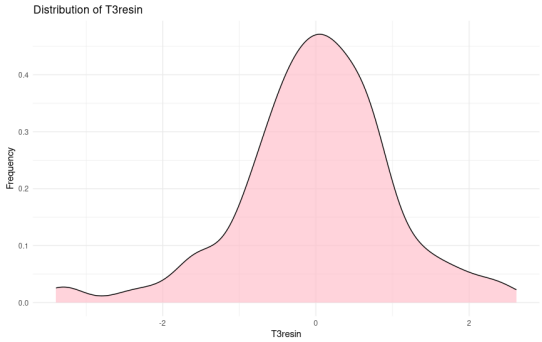

Figura 20: T3resin.

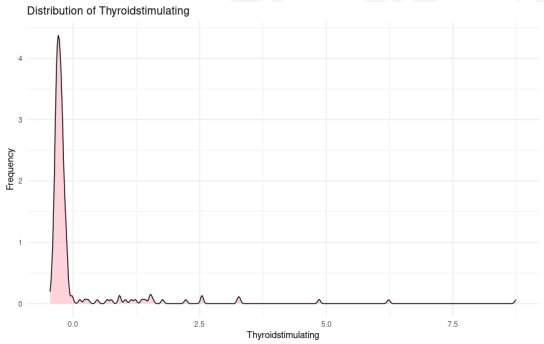

Figura 21: Thyroidstimulating.

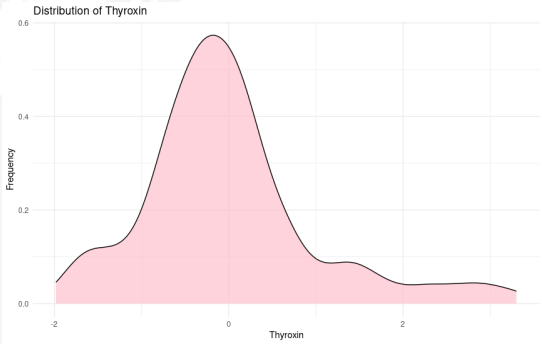

Figura 22: Thyroxin.

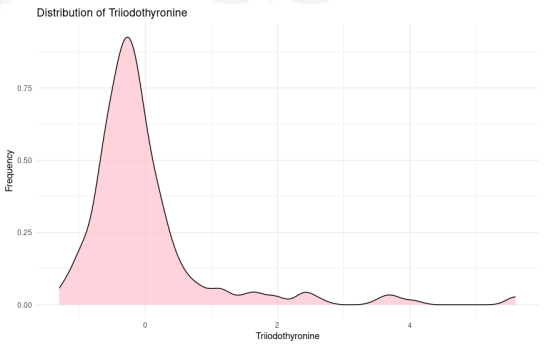

Figura 23: Triiodothyronine.

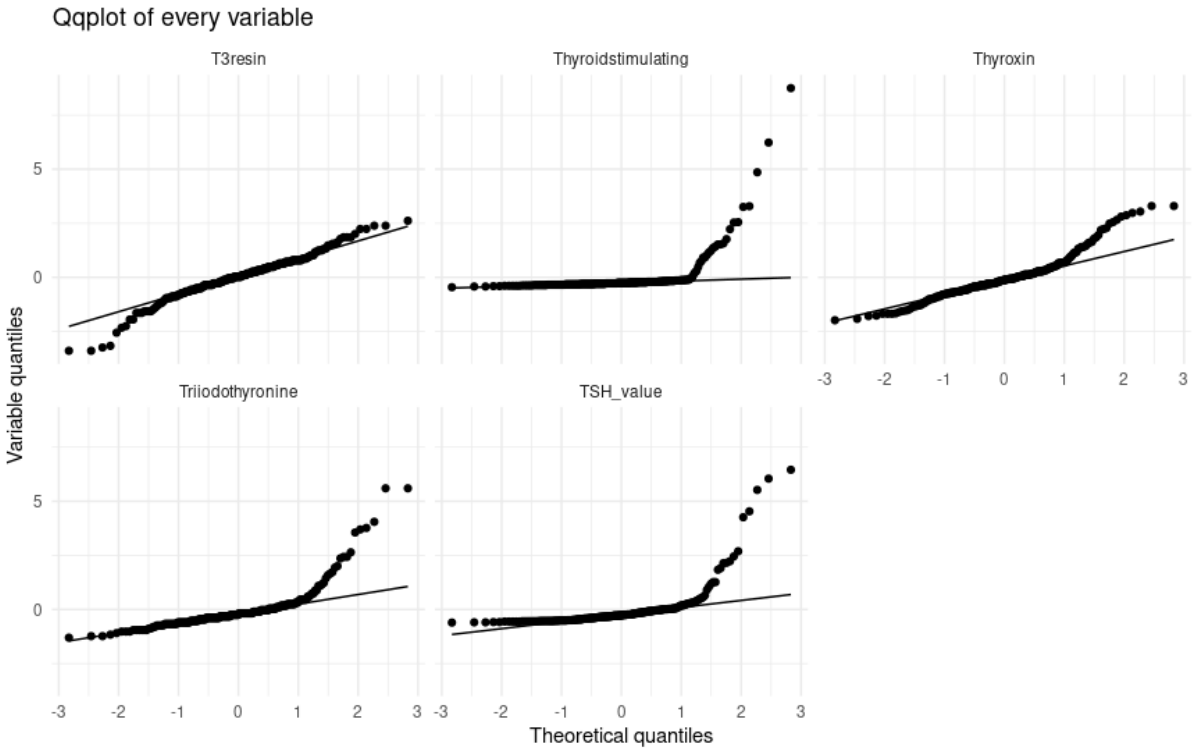

Figura 24: Qqplot for classification EDA.

En la figura 24 se puede ver más claro como las colas de las distribuciones son bastante grandes. En la mayoría, se presenta una asimetría a la derecha.

Si se les hace el test de *Agostino* (para la asimetría) se rechaza y por tanto no son normales y tienen alto grado de asimetría.

3.3. Outliers

En la sección de regresión se analizó tanto anomalías univariantes (en una sola característica) como variables multivariante. Se analizarán solo los posibles *outliers* univariantes, ya que si las variables por separado son uniformes, no lo serán en conjunto y por tanto métodos como *Mahalanobis* no serían tan efectivos (aunque podría usarse *LOF*, pero se excede de lo que ha de realizarse en esta asignatura). La normalidad multivariante implica la normalidad de las distribuciones marginales, por ejemplo, de las distribuciones univariantes de sus componentes [1].

En la figura de *boxplots* 25 se puede observar como hay variables (sobre todo *Thyroidstimulating*) que tienen valores muy alejados de la media. Aplicando la distancia intercuartil se obtiene que el 8 % de los datos en el conjunto son anómalos. El problema con la distancia intercuartil es que no es adecuada en distribuciones muy asimétricas, como es nuestro caso. Además, el estudiante no dispone de conocimiento experto para

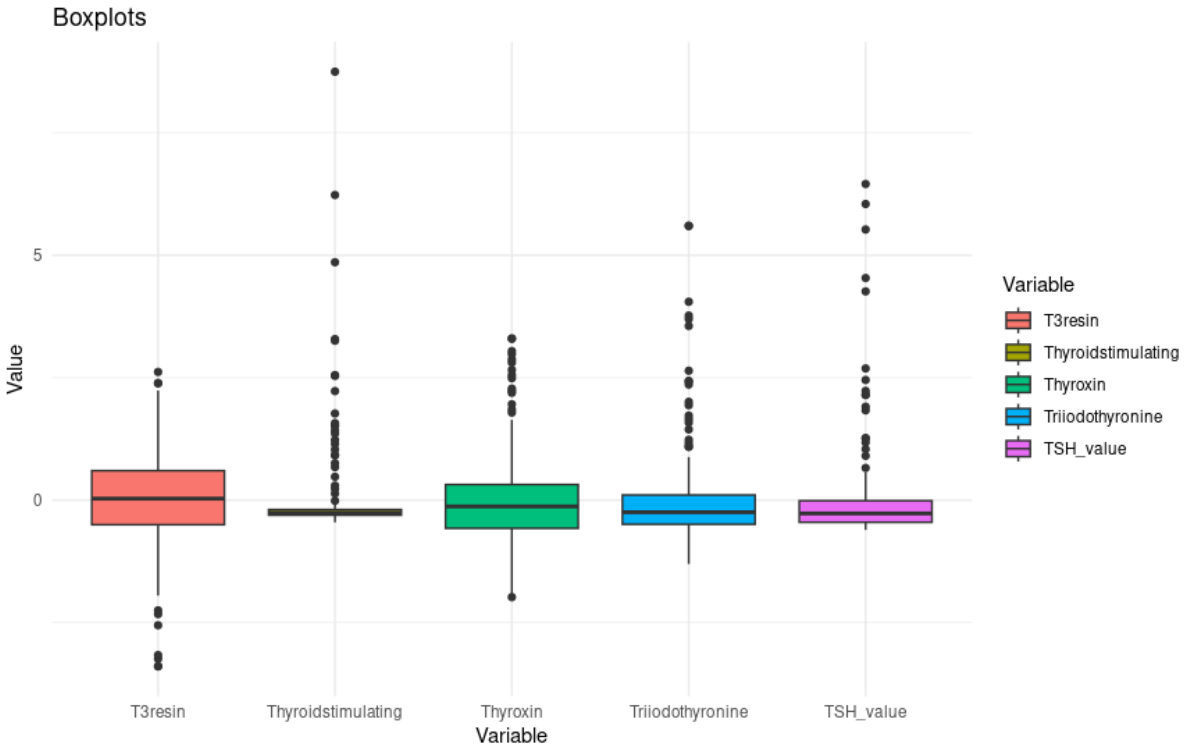

Figura 25: Boxplot for newthyroid.

valorar si las observaciones clasificadas como anómalas por este método son en realidad *outliers*. Por estas razones se decide dejar los valores.

3.4. Distribución por clases

Anteriormente se expuso que el conjunto de datos sufría de una desbalance de clases, por ello es interesante observar como se distribuyen las observaciones pertenecientes a cada clase en cada característica.

Dadas las gráficas de densidad por clase en 26,27,28,29,30 se observa como para las variables *T3resin*, *Thyroxin* y *Triiodothyronine* se suele clasificar en la clase 2 para valores más bajos y algo más dispersos. Valores altos de estas variables se relacionan más con las clases 1 y 2. Las distribuciones de clase para estas variables son muy parecidas, en cambio, en la variable *Thyroidstimulating* se puede percibir que las distribuciones son más discriminatorias para las clases 1 y 2, esto es interesante de cara a saber qué podría afectar a que se diese una clasificación u otra, ya que normalmente se observa que las clases 1 y 2 o crecen juntas o decrecen juntas mientras la clase 3 hace lo contrario. En esta variable se da un comportamiento nuevo.

Otra forma de observar la contribución de las variables a la clasificación es con un **Biplot** 31. En este se observa mejor la contribución de las variables entre sí y no por

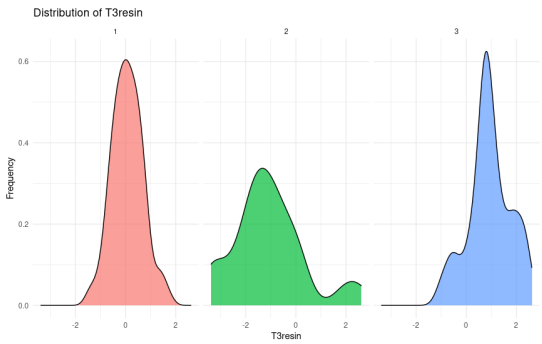

Figura 26: Class distribution for T3resin.

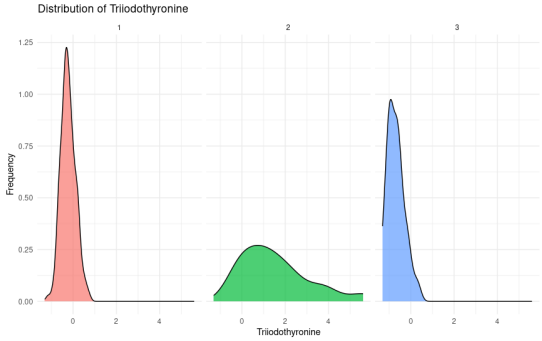

Figura 27: Class distribution for Triiodothyronine.

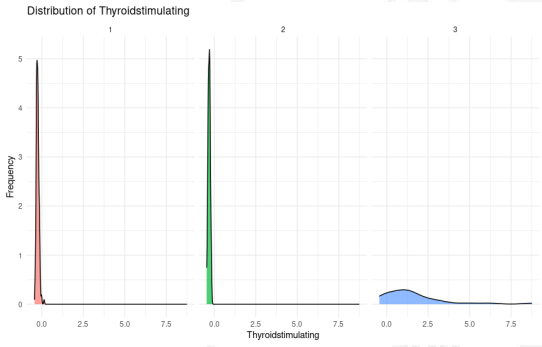

Figura 28: Class distribution for Thyroidstimulating.

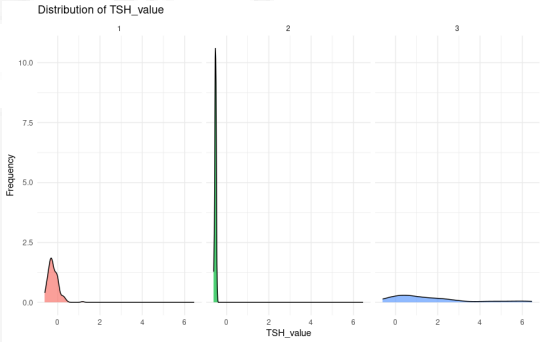

Figura 29: Class distribution for $\mathrm{TSH}_{v}alue$.

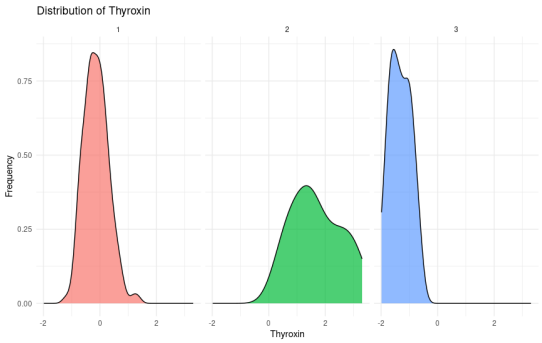

Figura 30: Class distribution for Thyroxin.

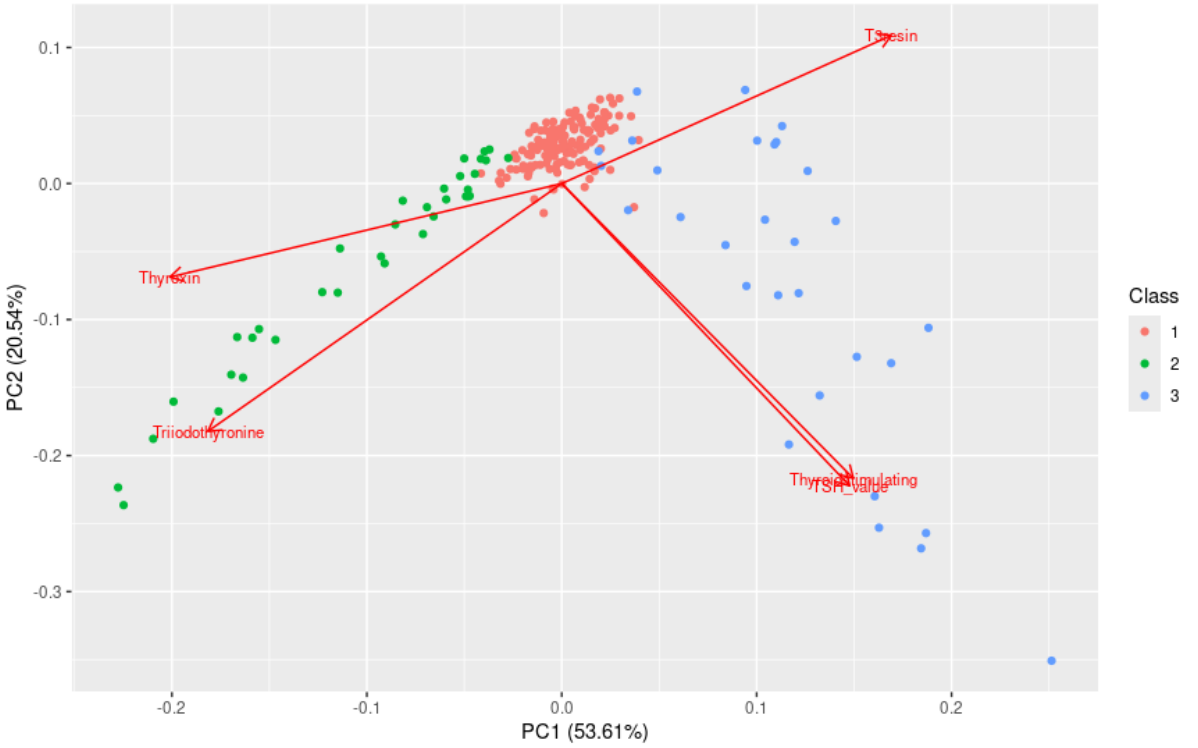

Figura 31: Biplot for classification EDA.

separado, como se ha analizado anteriormente. Combinaciones de valores altos en *Thyroxin* y *Triiodothyronine* parecen ser fuertes claves para que la observación sea clasificada como clase 2, por ejemplo.

3.5. Correlación

En este apartado se van a analizar las posibles relaciones entre variables del conjunto de datos. Para esto se pueden analizar gráficos visuales de puntos de variable *vs* variable y tablas de correlación.

Como puede observarse en el gráfico 32, existen algunas relaciones ligeramente lineales entre algunas variables, como es el caso de las variables *Thyroxin* y *Triiodothyronine*. El resto son menos obvias y no muy lineales. Si se obtiene la gráfica de la tabla de correlación, fig 16, el resultado que era obvio visualmente queda confirmado por un valor de correlación de 0,72.

Dados estos resultados, no se elimina ninguna característica por redundancia. Además, la dimensión del problema no es tal como para tener que optimizar el tamaño de las componentes/características que se tienen, pues hay muy pocas. De hecho muchas veces, dos características muy correladas no implican redundancia. Un ejemplo de esto aparece en [2] donde dos características con un ratio de correlación alto son totalmente necesarias

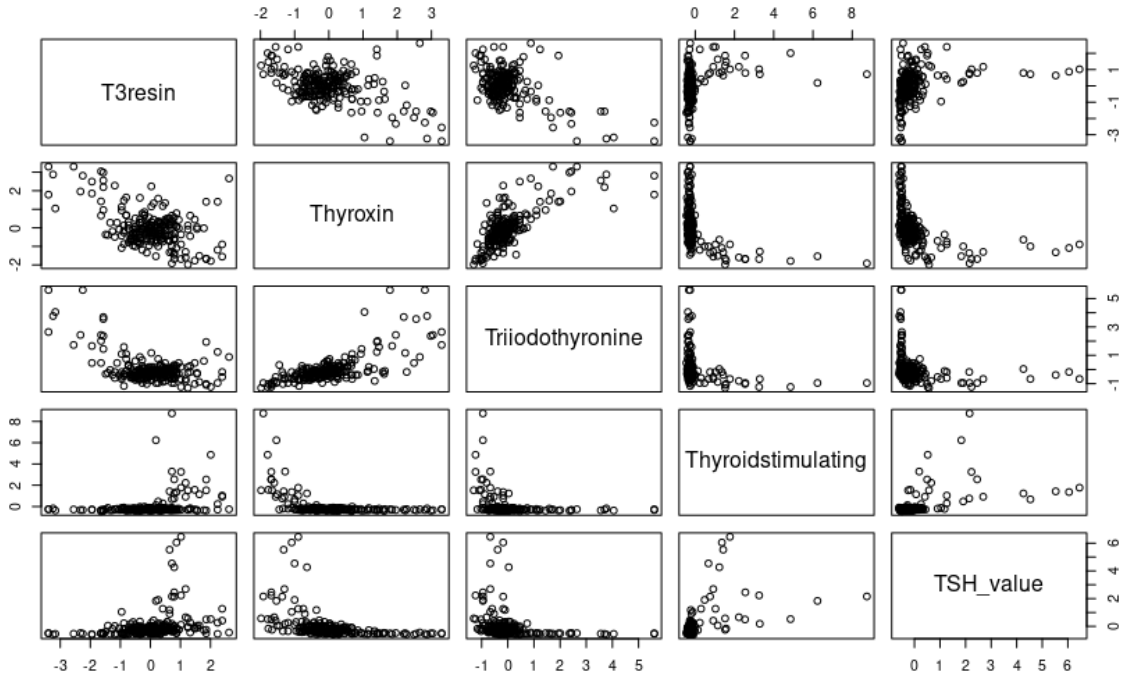

Figura 32: Pairs plot EDA classification.

(ambas) para la clasificación del problema.

3.6. Transformaciones

Si bien la normalidad es complicada alcanzarla en las variables que hay en el conjunto de datos (debido a grandes asimetrías) es posible intentar corregir algunas. Al aplicar la transformación logarítmica, los valores extremadamente altos se "comprimen", reduciendo el impacto de los *outliers* positivos y acercando la distribución de los datos a una forma más simétrica, pero no garantiza la normalidad. Además, en clasificación, las clases pueden volverse más separables en el espacio de características.

Se ha optado por transformar tres variables, aquellas más alejadas a las normales, como prueba para ver si se reduce la asimetría. Aplicando los test correspondientes se observa una reducción de la asimetría muy notable, aunque eso no hace que las distribuciones de las variables sean normales.

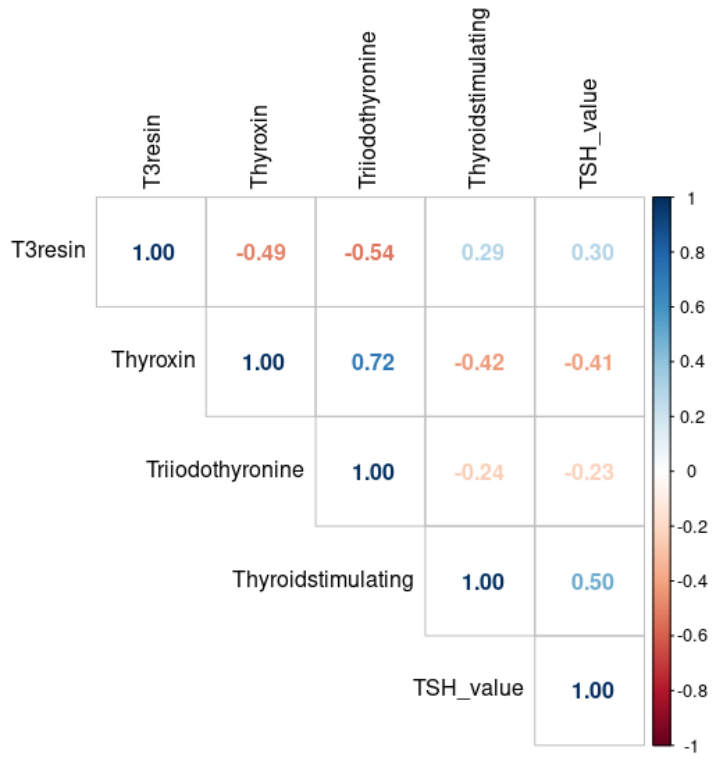

Figura 33: Correlation plot EDA classification.

4. Regresión

Se procede a describir el trabajo realizado en la parte de regresión con el conjunto de datos *wankara*.

4.1. Regresión simple

En este apartado se describirá el proceso que se ha seguido para obtener 5 regresores y cuál ha sido seleccionado como el mejor.

En el apartado del *EDA* de regresión se explicó que este conjunto de datos tiene 9 características, por lo que se debe elegir entre el total los que parezcan más prometedores. Según lo visto en el *EDA*, en lo relativo a correlación y al propio sentido común, la variable más prometedora es *diff_temperature* (es la combinación lineal de *max_temperature* y *min_temperature*). Seguido de *dewpoint*, *diff_pressure* (combinación lineal de variables originales de presión), *visibility* y *wind_speed*.

Se ha tenido en cuenta las variables que parecen más lineales con respecto a la variable objetivo, o que al menos parezcan tener una relación en algún sentido (por ejemplo, *visibility* es algo curva, algo logarítmica).

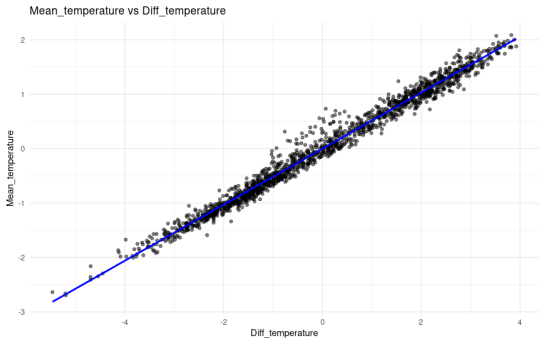

Figura 34: Diff temperature vs Mean Temperature.

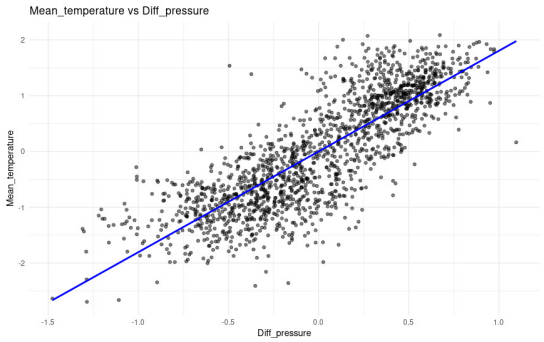

Figura 35: Diff pressure vs Mean Temperature.

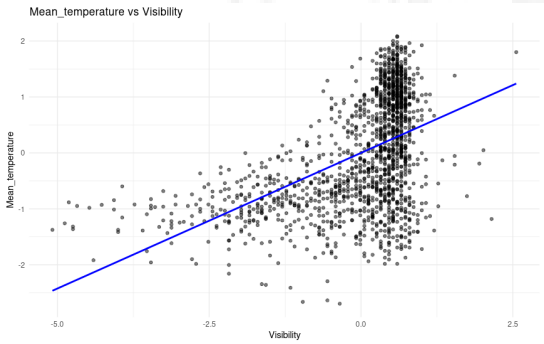

Figura 36: Visibility vs Mean Temperature.

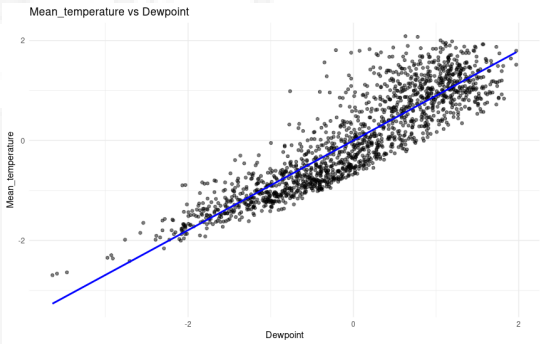

Figura 37: Dewpoint vs Mean Temperature

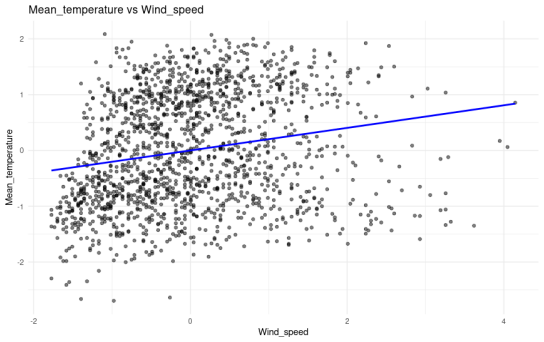

Figura 38: Wind speed vs Mean Temperature.

Cuadro 4: Linear Simple Regression Results

	Residual Std. Error	t value	p-value	R^2
Diff_temperature	0.132	301.2	$< 2 \times 10^{-16}$	0.9826
Dewpoint	0.4433	81.09	$< 2 \times 10^{-16}$	0.8036
Diff_pressure	0.5656	58.47	$< 2 \times 10^{-16}$	0.6803
Visibility	0.8749	22.22	$< 2 \times 10^{-16}$	0.235
Wind_speed	0.9796	8.29	$2{,}37 \times 10^{-16}$	0.0410

Cuadro 5: Summary of simple linear regression.

Como puede observarse en la tabla 5 la mejor variable es sin duda alguna *diff_temperature*, obtiene un error mucho mejor y el valor R^2 es casi de 1, lo que indica que prácticamente solo con esa variable se explica la varianza total de la variable objetivo. Se añaden gráficas de cada variable por separado y como estas se relacionan con la variable objetivo (figuras 34,35,36,37,38).

4.2. Regresión múltiple

En esta aparado se consideran múltiples variables así como las interacciones entre ellas y posibles no linealidades para la mejora del modelo.

4.2.1. Backward selection

Inicialmente se utiliza la técnica de selección de características *backward*. Esta consiste en un enfoque iterativo que comienza con el conjunto completo de características disponibles y elimina progresivamente aquellas que son menos relevantes para el modelo (por medio del *p-valor*), hasta que se alcanza un subconjunto "óptimo"de características.

Realizando este proceso se eliminan las variables *max_wind_speed* y *precipitation*. El resto de variables obtienen valores de *p-valor* por debajo de 0,05, por lo que con un 95 % de confianza se podría decir que están aportando suficiente al problema.

4.3. Interacciones y no linealidad

Se realizan pruebas añadiendo variables. Si bien es cierto que en la selección hacia atrás solo se eliminaron dos variables, como se ha analizado anteriormente solo *diff_temperature* parece ser suficiente, por ello se prueba primero esa variable aislada y luego se añaden más.

```
fit1 <- data %>%
  lm(formula = Mean_temperature ~ Diff_temperature, data = .)

```

```
summary(fit1)


fit2 <- data %>%
  lm(formula = Mean_temperature ~ Diff_temperature + Dewpoint + Diff_
    pressure,
      data = .)

summary(fit2)

fit3 <- data %>%
  lm(
    formula = Mean_temperature ~ Diff_temperature + Dewpoint + Diff_
    pressure + Visibility,
     data = .
  )

summary(fit3)
```

El resultado de ello es que no se mejora apenas. Se prueban interacciones entre variables con sentido, como puede ser *dewpoint* o punto de condesación y la variable de la temperature. Se consideran también interacciones no lineales entre estas dos variables.

```
fit4 <- data %>%
  lm(formula = Mean_temperature ~ Diff_temperature * Dewpoint,
     data = .)

summary(fit4)

fit5 <- data %>%
  lm(formula = Mean_temperature ~ Diff_temperature * I(Dewpoint ^ 2),
     data = .)

summary(fit5)

fit6 <- data %>%
  lm(formula = Mean_temperature ~ Diff_temperature * I(Dewpoint ^ 2) *
    Dewpoint,
      data = .)

summary(fit6)

fit7 <- data %>%
  lm(
    formula = Mean_temperature ~ Diff_temperature * I(Dewpoint ^ 2) *
    Dewpoint * I(Diff_temperature ^

                   2),
    data = .
  )

summary(fit7)
```

Las métricas mejoran, se pasa de un error estándar de 0,13 a 0,11. Aunque es poco, es el mejor resultado hasta el momento.

Se han probado algunas combinaciones más, pero no tienen demasiado sentido (a priori) y no parecen mejorar en absoluto el modelo. También hay que tener en cuenta que un ajuste demasiado fino puede resultar en un **sobreajuste**.

4.4. KNN

Se realizan pruebas igual que con el algoritmo de regresión lineal, pero usando *KNN*. En este caso se obtienen resultados (a priori) mejores que en regresión lineal.

4.5. Comparativas

En esta sección se realizan comparativas entre los algoritmos de regresión lineal múltiple, *KNN* y *M5*. Para ello se implementa una función de *k-fold cross validation* con $k = 5$. Para una comparación justa entre algoritmos se ajustan los modelos con todas las variables.

Algorithm	**Mean accuracy**
Multiple Linear Regression	0.01034
KNN	0.01777
M5	0.02576

Cuadro 6: Summary of results in k-fold validation.

Como puede observarse en la tabla 6, el algoritmo de regresión lineal ha obtenido mejores resultados que los otros dos. Esto se debe a que la naturaleza de este problema es lineal y por tanto el algoritmo que mejor se ajusta a esto es el de regresión lineal. La temperatura media tiene relaciones lineales con algunas variables como son temperatura máxima, mínima y punto de condensación. Existen otras, pero las variables más importantes, como se analizó anteriormente, son las mencionadas, y sus correlaciones con la variable objetivo son casi perfectas.

Se muestran gráficas de las predicciones en cada *fold* de los diferentes algoritmos en 39,40,41. Los valores que más se acercan son los del modelo de regresión lineal, pero los otros algoritmos obtienen un ajuste muy bueno igualmente. El algoritmo de regresión lineal obtiene mejor varianza con respecto a las observaciones actuales y las observaciones predichas.

Para las comparaciones entre algoritmos con test estadísticos es necesario realizar ciertas normalizaciones, ya que en regresión los errores no están en las mismas escalas. Se usa la fórmula 1.

Para comparar los algoritmos, se realizan los test de *Wilcoxon*. El test de *Wilcoxon* es una

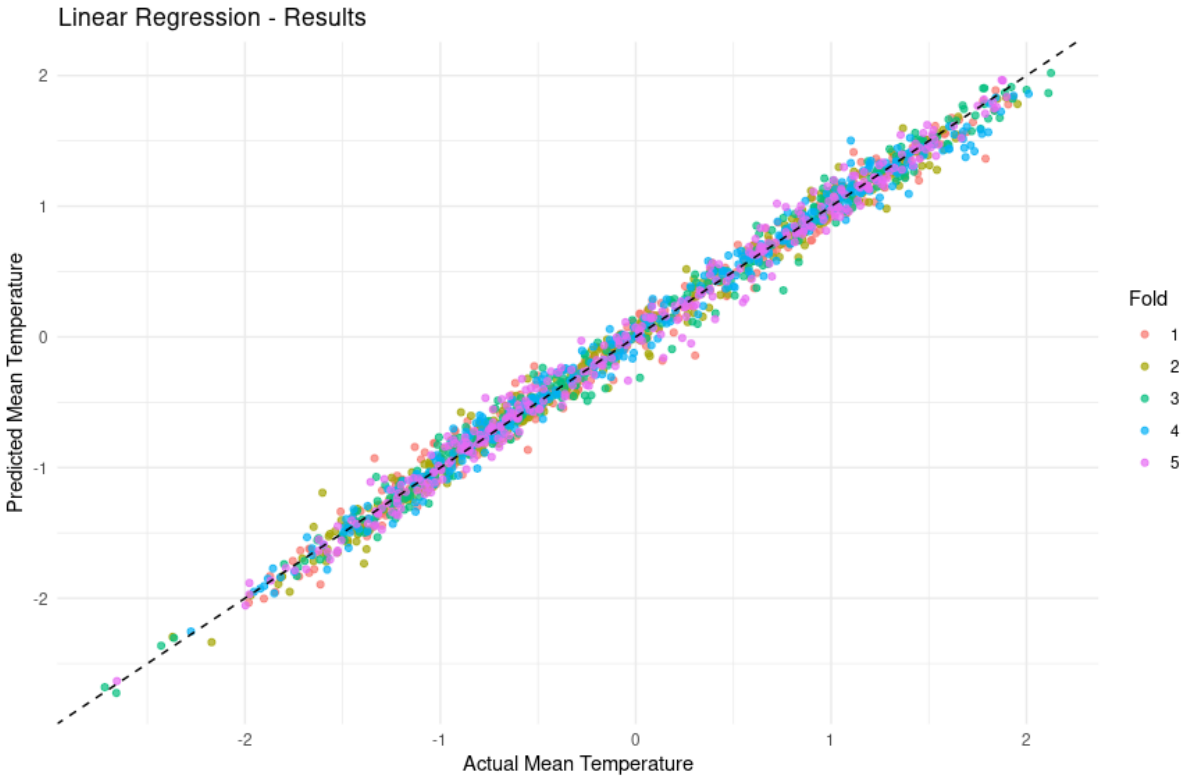

Figura 39: K-fold cross validation results in linear regression.

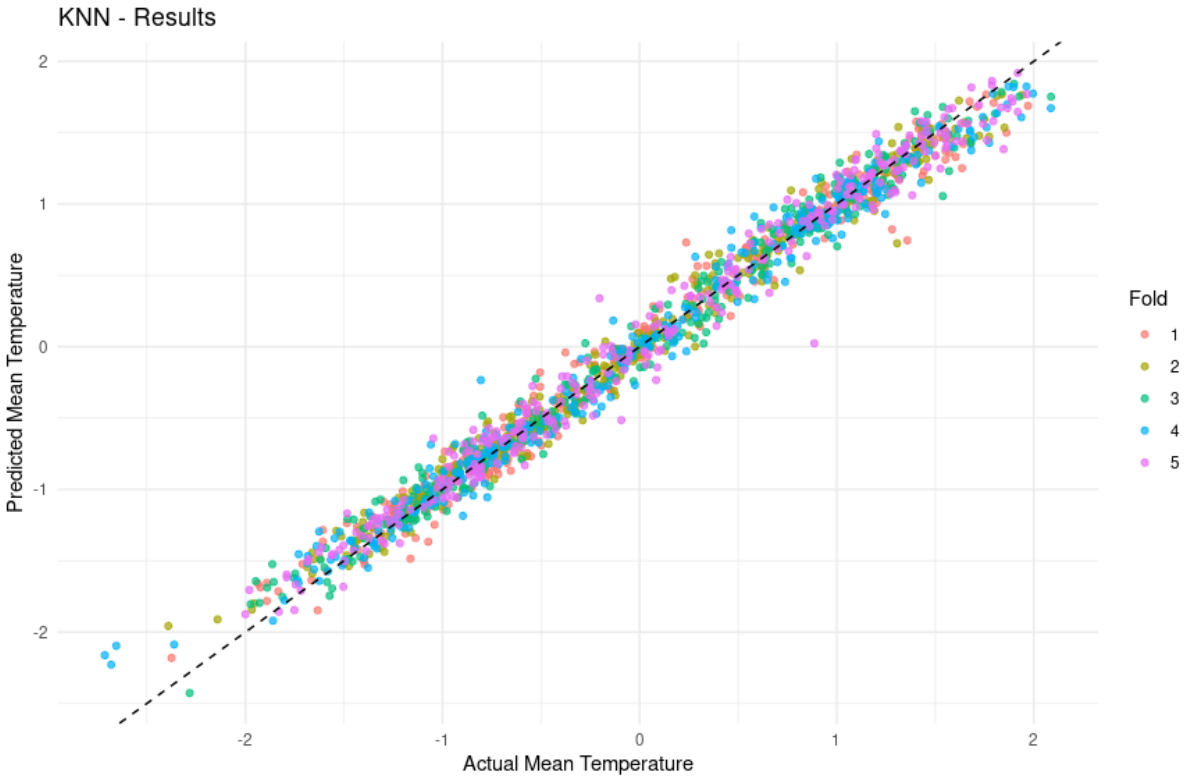

Figura 40: K-fold cross validation results in **KNN**.

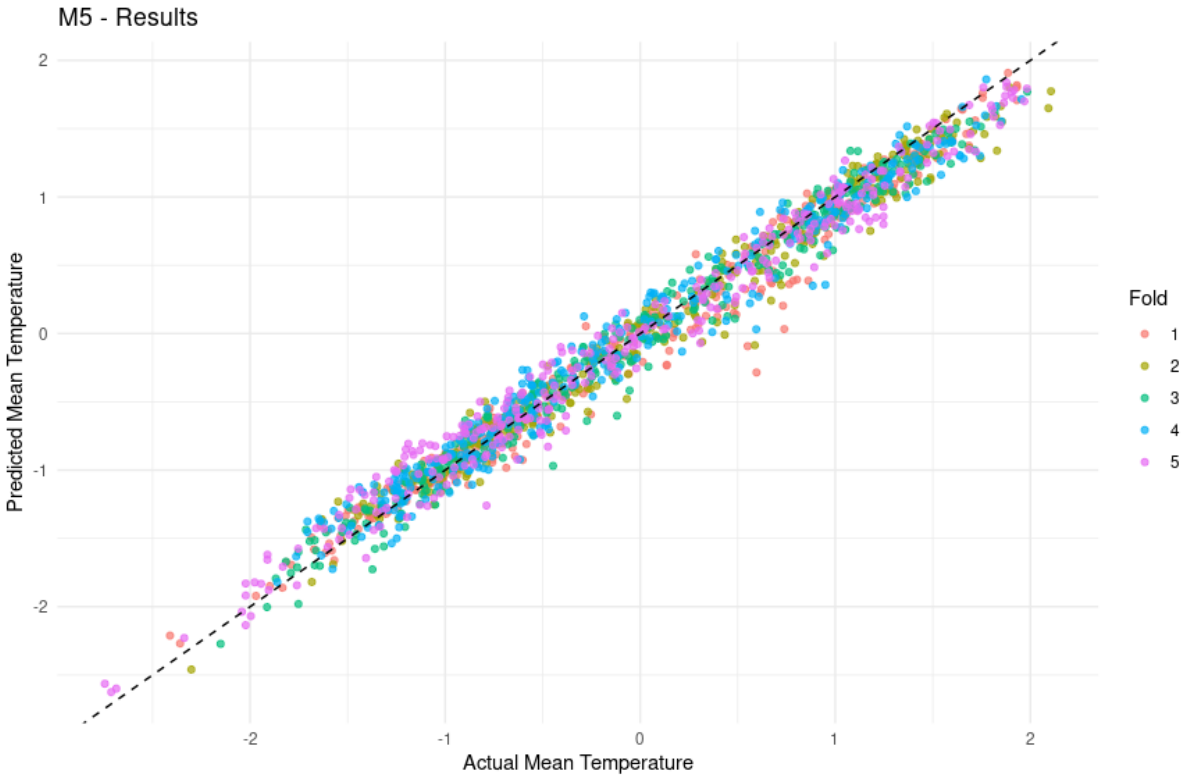

Figura 41: K-fold cross validation results in **M5**.

prueba estadística no paramétrica utilizada para comparar dos conjuntos de datos emparejados y determinar si sus distribuciones difieren significativamente. Se utiliza cuando no se puede asumir que los datos siguen una distribución normal.

$$\text{DIFF} = \frac{\text{Mean(Other)} - \text{Mean(Reference Algorithm)}}{\text{Mean(Other)}} \tag{1}$$

Los resultados en los test son los obtenidos en 2. Como puede verse, comparando los algoritmos de **KNN** vs los de regresión lineal múltiple, no se obtienen resultados estadísticamente significativos. Por ello, no se puede decir que un algoritmo sea mejor que otro, ya que las mejoras podrían haberse dado por pura aleatoriedad. No hay suficiente evidencia.

$$\text{p-value} = 0{,}701, \quad \text{R+} = 76, \quad \text{R-} = 95 \tag{2}$$

Para comparar los tres algoritmos es necesario hacer el test de **Friedman** y realizar una corrección *post-hoc*, ya que solo con **Friedman** se acumulan errores que pueden invalidar las conclusiones. Se utiliza **Friedman** ya que es una alternativa no paramétrica al **ANOVA** de medidas repetidas, pues no asume normalidad en los datos.

Tras la corrección se obtienen los resultados en la tabla 7.

	1	2
2	0.44	-
3	0.16	0.28

Cuadro 7: Friedman test + Holms (1: Linear Regression, 2: KNN, 3: M5) en *test*.

Tampoco se obtienen resultados significativos.

Ahora se realizan las mismas comparaciones para el conjunto de *training*.

$$\text{p-value} = 0{,}00025, \quad \text{R+} = 9, \quad \text{R-} = 162 \tag{3}$$

	1	2
2	0.00057	-
3	0.00475	0.00107

Cuadro 8: Friedman test + Holms (1: Linear Regression, 2: KNN, 3: M5) en *training*.

En las tablas 3, 8 se obtienen resultados estadísticamente significativos. De hecho si se comparan las medianas, se obtienen resultados a favor de los algoritmos **M5** y **KNN** frente a regresión lineal. Concretamente, la mediana para **M5** en *train* es de 2,8, la de **KNN** es de 1,8 y regresión lineal de 5. Por ello, las diferencias son significativas para cada comparación entre pares y concretamente, **KNN** sale parado como el mejor (en *train*). En *test* sin embargo, obtiene mejor mediana **M5**, pero los test no rechazan.

El hecho de que los test rechacen en el conjunto de entrenamiento y no en el de evaluación puede significar un posible sobreajuste. Si los resultados no son significativos en *test*, significa que los modelos no logran reproducir esos mismos resultados con los datos no vistos.

5. Clasificación

Se procede a describir el trabajo realizado en la parte de clasificación con el conjunto de datos *newthyroid*.

5.1. KNN

Se realiza un pequeño estudio para comprobar que valor para k es mejor en este problema concreto. Para ello se han aplicado las transformaciones necesarias definidas en el **EDA** de clasificación y se ha definido una función de particionamiento en conjuntos de datos de entrenamiento y de evaluación. Se ha omitido transformar la variable de *Triiodothyronine* con el logaritmo ya que esta transformación produce muchos nulos (hay

valores no definidos para el logarítmo dentro de la variable). El resto de transformaciones logarítmicas se realizan para reducir asimetrías.

K	**Accuracy**
1	0.95
5	0.95
15	0.93
40	0.81
100	0.7

Cuadro 9: K values and accuracies for **KNN**.

Como puede observarse en la tabla 9, se obtienen muy buenos resultados para valores de k pequeños, incluso para $k = 1$, que es un valor muy dado al sobreajuste debido a que hace al modelo mucho más sensible a cambios cuando se añaden nuevos datos, añade mucha varianza. Para valores más seguros, como $k = 5$ los resultados siguen siendo muy buenos. Para valores superiores a 15, el modelo tiende a equivocarse cada vez más, está aumentando el sesgo del modelo y por tanto generaliza cada vez peor.

Se añaden las gráficas de matrices de confusión (fig 42,43,44,45,46) donde puede verse que los primeros modelos con k bajos ajustan a la perfección las clases predichas y mientras este valor de k aumenta, suele clasificar erróneamente con la clase 1. Según se explicó en el **EDA**, había un gran desbalance con la clase 1 frente a las otras dos, por lo que es normal que según el modelo sea peor, este tienda a clasificar con la clase más mayoritaria.

Se escoge finalmente el modelo con $k = 5$ debido a que tiene una métrica de precisión igual y se reduce la complejidad del modelo.

5.2. LDA

Se procede a realizar un ajuste de los datos con el algoritmo **LDA** (*Linear Discriminant Analysis*). **LDA** es una técnica supervisada utilizada principalmente para reducción de dimensionalidad y clasificación. Funciona buscando maximizar la separabilidad entre clases proyectando los datos en un espacio de menor dimensión. Este algoritmo asume ciertas propiedades sobre los datos, por ello, es necesario realizar una serie de comprobaciones sobre los datos.

5.2.1. Asunciones

El algoritmo **LDA** asume que:

- Cada característica del conjunto de datos es normal para cada clase, es decir, los datos dentro de cada clase forman una campana simétrica alrededor de la media.

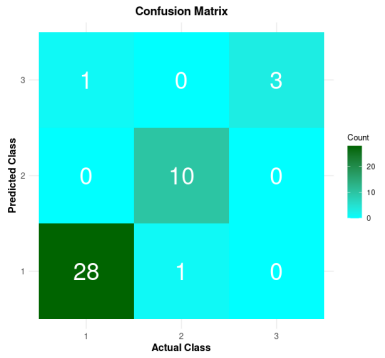

Figura 42: confusion Matrix for $k = 1$.

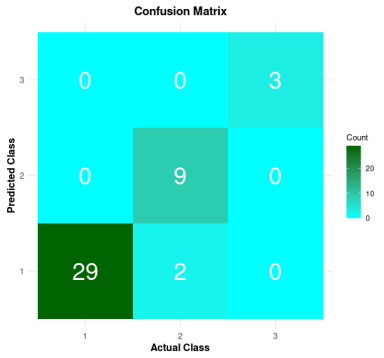

Figura 43: confusion Matrix for $k = 5$.

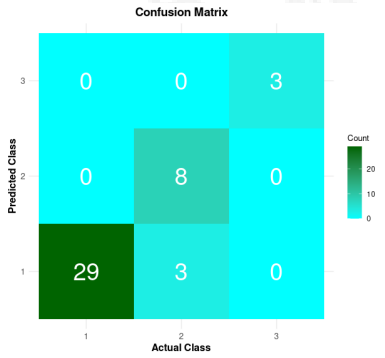

Figura 44: confusion Matrix for $k = 15$.

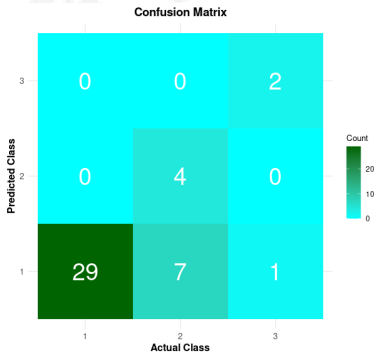

Figura 45: confusion Matrix for $k = 40$.

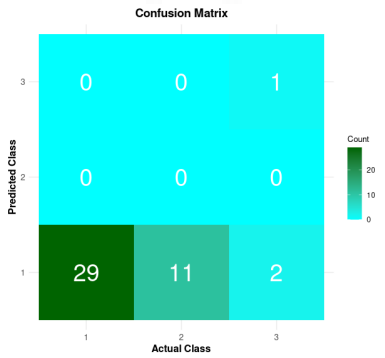

Figura 46: confusion Matrix for $k = 100$.

- Se asume que la matriz de covarianza de las características es la misma para todas las clases. Esto implica que las clases tienen la misma dispersión y varianza en todas las dimensiones del espacio de características.
- Los datos obtenidos para la creación del *dataset* se han obtenido como una muestra aleatoria.

Para comprobar la primera asunción se realiza una serie de test sobre cada variable agrupada por clase, concretamente el test de *Shapiro*. Se utiliza la siguiente función, la cual también crea una gráfica *QQPlot*.

```
test_variable <- function(var_name) {
  resultados <- data %>%
    group_by(Class) %>%
    summarize(
      p_value = if (is.numeric(.data[[var_name]]) &&
                    all(!is.na(.data[[var_name]]))) {
        shapiro.test(.data[[var_name]])$p.value
      } else {
        NA
      },
      normal = ifelse(p_value > 0.05, "Puede ser normal", "No es normal
   "),
      .groups = "drop"
    )
  plot <- ggplot(data, aes(sample = .data[[var_name]])) +
    stat_qq() + stat_qq_line() +
    facet_wrap(~ Class)
  print(plot)
  print(resultados)
}
```

Como muestra la tabla 10, algunas variables rechazan el test, por lo que no pueden ser consideradas normales con una garantía estadística del 95 %.

Para comprobar la igualdad de matrices de covarianza-varianza se realizan test de *Bartlett* para cada variable. Los resultados son que todas las variables rechazan con una diferencia estadística muy significativa, lo que significa que las varianzas de los grupos no son iguales y, por lo tanto, no se cumple el supuesto de homocedasticidad (es decir, las varianzas iguales entre grupos).

5.2.2. Ajuste

Al ajustar el modelo con **LDA** se obtiene que la primera variable discriminante aporta un 83 % de la variable explicada, mientras que la segunda aporta un 17 %. Esto indica que la primera variable es la que consigue la mayor parte de la separación. Dentro de los datos recogidos se observa que la variable que más aporta a **LD1** es *TSH_value* con un coeficiente de 1,42.

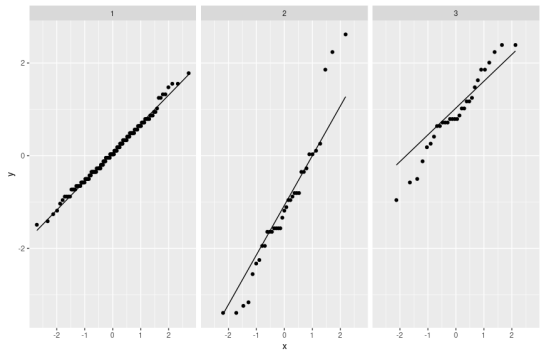

Figura 47: **QQPlot** for T3resin grouped by class.

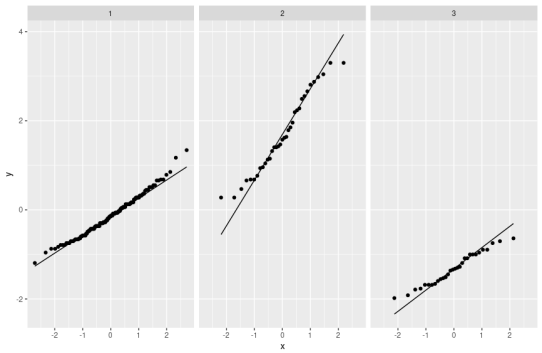

Figura 48: **QQPlot** for Thyroxin grouped by class.

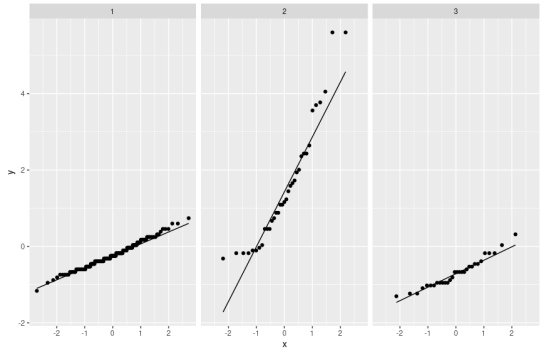

Figura 49: **QQPlot** for Triiodothyronine grouped by class.

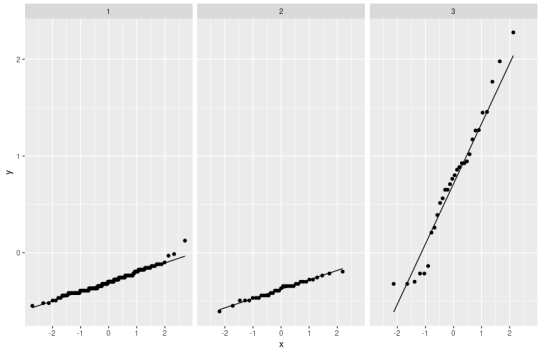

Figura 50: **QQPlot** for Thyroidstimulating grouped by class.

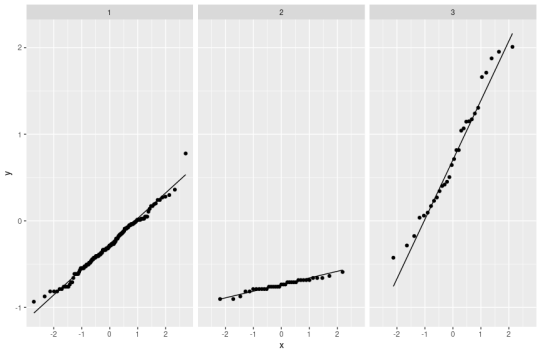

Figura 51: **QQPlot** for TSH_value grouped by class.

Variable	Class	p_value	normal
T3resin	1	0.736	Puede ser normal
	2	0.0479	No es normal
	3	0.414	Puede ser normal
Thyroxin	1	0.143	Puede ser normal
	2	0.200	Puede ser normal
	3	0.448	Puede ser normal
Triiodothyronine	1	0.257	Puede ser normal
	2	0.00357	No es normal
	3	0.189	Puede ser normal
Thyroidstimulating	1	0.00683	No es normal
	2	0.865	Puede ser normal
	3	0.352	Puede ser normal
TSH_value	1	0.251	Puede ser normal
	2	0.293	Puede ser normal
	3	0.405	Puede ser normal

Cuadro 10: Normality test for every variable grouped by class.

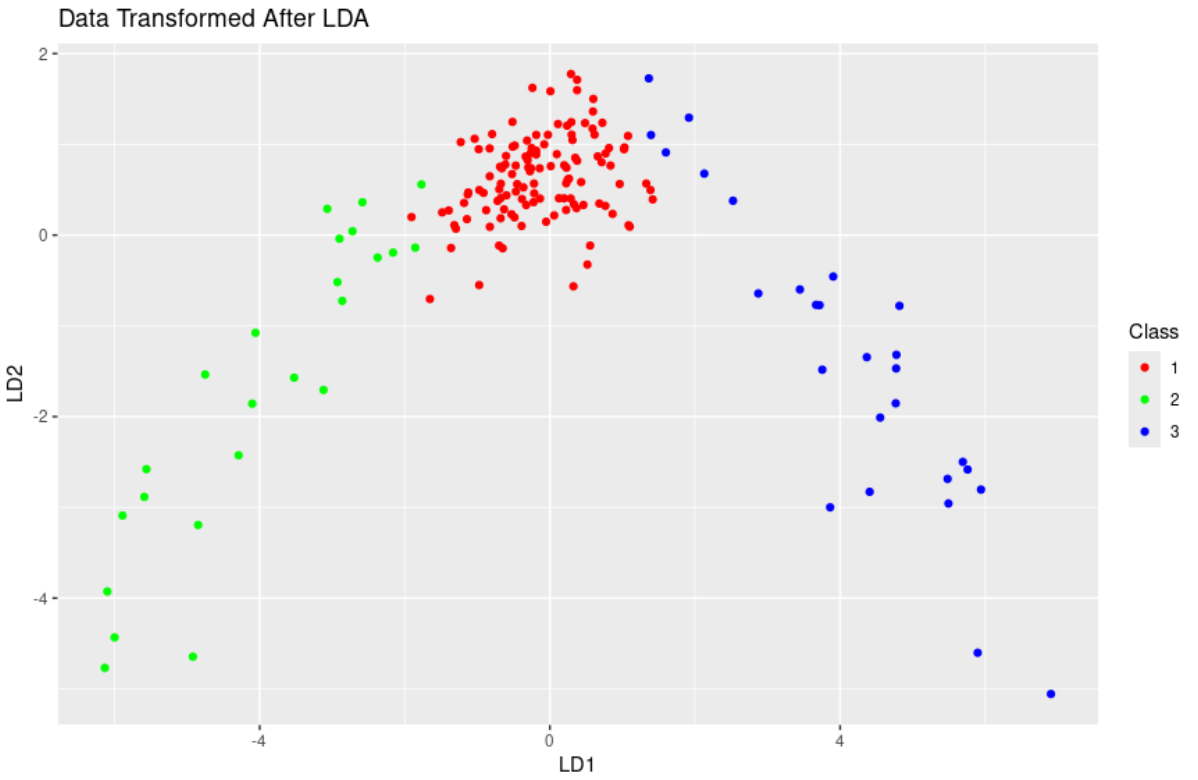

Figura 52: Data transformation after **LDA**.

En la figura 52 se puede ver como se transforman los datos de *train* tras aplicar el algoritmos. Se obtiene un *accuracy* del 90,6 %. Pese a no cumplir la segunda asunción y la primera en parte, el algoritmo es capaz de ajustar los datos muy bien.

5.3. QDA

A diferencia de **LDA**, **QDA** no asume que todas las clases compartan la misma matriz de covarianza. Cada clase puede tener su propia matriz de covarianza. Como consecuencia, **QDA** genera fronteras de decisión cuadráticas (curvas) entre los grupos. Esto permite que **QDA** sea más flexible, ya que puede modelar relaciones no lineales entre las variables.

A priori, se podría suponer que este algoritmo debería ir mejor que **LDA** ya que ninguna de las variables cumplía la segunda asunción. De hecho, al ajustar el modelo, se obtiene un 97 % de precisión. En la matriz de confusión (fig 53) puede observarse la frecuencia de clasificación por clases.

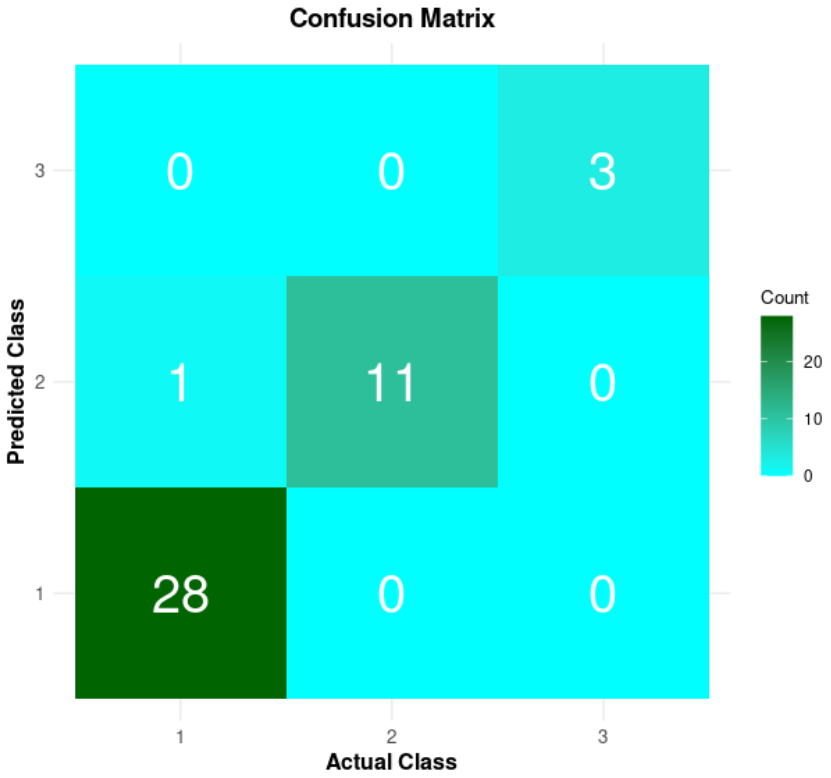

Figura 53: Confusion Matrix for **QDA**.

5.4. Comparación

Al realizar una comparación de los tres algoritmos usando 10-*fold cross validation*, se obtienen los resultados descritos en la tabla 11. **KNN** obtiene unos resultados muy

buenos. **LDA** obtiene una precisión muy alta, pero se queda por detrás de los otros dos algoritmos, además es más variante en comparación a **KNN** y **QDA** (aunque sigue siendo un valor de desviación típica muy bajo). Es de esperar que este algoritmo fuese a ser el peor, pues sus asunciones básicas no eran cumplidas en algunos casos. Pese a ello los resultados siguen siendo muy buenos.

El algoritmo que mejores resultados a logrado para este conjunto de datos es **QDA**, con un asombroso 97 % de precisión y la desviación típica más baja.

Algorithm	**Mean accuracy**
KNN	$0{,}96 \pm 0{,}039$
LDA	$0{,}933 \pm 0{,}053$
QDA	$0{,}976 \pm 0{,}033$

Cuadro 11: Summary of results in k-fold validation.

6. Apéndice

6.1. EDA Regresión

```
library(tidyverse)
library(moments)
library("corrplot")
library(ggfortify)
library(MVN)

"Leemos los datos saltandonos las cabeceras iniciales y despu s las parseamos
a mano. Seguido, vamos a ver unos pocos datos con head para hacernos una primera
idea."

data <- read.csv("wankara/wankara.dat", skip = 14, header = FALSE)
colnames(data) <- c(
  "Max_temperature",
  "Min_temperature",
  "Dewpoint",
  "Precipitation",
  "Sea_level_pressure",
  "Standard_pressure",
  "Visibility",
  "Wind_speed",
  "Max_wind_speed",
  "Mean_temperature"
)
head(data)

"Con la funci n de summary obtenemos un resumen estad stico general de cada
```

```
columna. Adem s vamos a obtener otros datos informativos como la dimensi n ,
y la estructura."

summary(data)
dim(data)
str(data)
colSums(is.na(data))

"Los datos parecen bastante buenos a priori. No hay escalas num ricas muy
grandes, no hay datos faltantes y en principio parece que las distribuciones
se centran en la media (a excepci n de algunas variables como Precipitation y
alguna m s). En principio lo m s intuitivo ser a analizar visualmente las
variables."

plot_distribution <- function(data,
                              var_name,
                              binwidth = 0.1,
                              fill_color = "pink") {
  ggplot(data, aes_string(x = var_name)) +
    geom_density(fill = fill_color,
                 color = "black",
                 alpha = 0.7) +
    labs(
      title = paste("Distribution of", var_name),
      x = var_name,
      y = "Frequency"
    ) +
    theme_minimal()
}

for (column in names(data)) {
  p <- plot_distribution(data, column)
  print(p)
}

"Las variables de temperatura m xima y m nima parecen adecuarse a la forma de
una pseudo-bimodal (Tienen dos picos y una depresi n claramente diferenciados).
La variable de precipitaci n es asim trica extrema, con valores pr cticamente
nulos (entiendase por nulo el cero). Las relativas a la presi n son
distribuciones que si bien no son normales exactas, se asimilan mucho (ambas
poseen la peculiaridad de un peque o \"bulto\"para presiones de entre 26-30. En
cuanto al resto, hay variedad de distribuciones con asimetr as."

```

```
"La bimodalidad en las temperaturas puede darse por varios motivos, pero el que
considero m s probable es la toma de datos en distintas zonas geogr ficas de
Ankara, dando lugar a distintas modas. El calentamiento clim tico podr a ser
factible, pero no ser a tan evidente y se necesitar an tomas de muchos a os."

"Vamos a escalar los datos, ya que as  se reduciran las diferencias entre
variables. Escalar permite que todas las variables contribuyan equitativamente a
la distancia calculada, evitando que una variable desproporcionada influya en
la detecci n de outliers."

data <- as.data.frame(scale(data))
summary(data)

"A continuaci n vamos a aplicar test de normalidad a aquellas variables m s
prometedoras y ver si la cumplen. Adem s obtenemos los qqplots de todas las
variables para ver como de cerca est n las distribuciones de la normal te rica."

long_data <- data %>%
  pivot_longer(
    cols = setdiff(names(data), "Mean_temperature"),
    names_to = "Variable",
    values_to = "Value"
  )

ggplot(long_data, aes(sample = Value)) +
  stat_qq() +
  stat_qq_line() +
  facet_wrap(~ Variable) +
  labs(title = "Qqplot of every variable", y = "Variable quantiles", x = "Theoretical quantiles") +
  theme_minimal()

shapiro.test(data$Sea_level_pressure)
shapiro.test(data$Standard_pressure)

"Vamos a analizar los boxplots de las variables"

ggplot(long_data, aes(x = Variable, y = Value, fill = Variable)) +
  geom_boxplot() +
  labs(title = "Boxplots", x = "Variable", y = "Value") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

```

```
"Observando las variables y sus valore m ximos y m nimos no considerar a que
existan outliers univariantes, ya que son valores posibles y nada irreales.
Las precipitaciones es la que tiene valores m s extremos, pero por lo general,
al ser una zona seca, es normal que la mayor a de valores sean cero. M s tarde
comprobar  si pueden existir outliers multivariantes."

"Si bien es cierto que el test rechaza (no es una distribuci n normal). Los
qqplot de las variables concernientes a la presi n son muy buenos. De hecho en
las gr ficas de densidad hemos observado que no presentan formas an malas. Dados
estos resultados, pese a no pasar los test, podr amos tomar una postura un poco
m s relajada en cuanto a restricciones de la normal en lo que se refiere a esas dos
variables, sobre todo en algunas t cnicas (si se llegan a usar) como ANOVA, que
son muy robustas a violaciones de normalidad.
El resto de variables son tambi n bastante normales (algo asim tricas).
Se dan excepciones, por ejemplo, La visibilidad tiene una cola muy larga a la
derecha, al igual que la m xima velocidad del viento (aunque ya hemos visto que
las variables de viento parecen bimodales)."

"Vamos a ver la correlaci n de las caracter sticas"

cor_matrix <- cor(data)
corrplot(cor_matrix,
         method = "number",
         type = "upper",
         tl.col = "black")

"Existen fuertes coeficientes de correlaci n entre variables de m nimo, m ximo y
temperatura media. Estas correlaciones son muy grandes y positivas, cuando crece
una variable lo hace la variable objetivo. Adem s el punto de condensaci n o
dewpoint tambi n tiene correlaci n con la variable objetivo. Estas correlaciones
tienen sentido. El hecho de que el punto de condensaci n o dewpoint tambi n est
correlacionado con la variable objetivo refuerza la idea de que las condiciones
atmosf ricas, como la humedad y la temperatura, est n relacionadas con el
```

```
comportamiento de la variable de inter s. En particular, el dewpoint es un
indicador importante de la humedad en el aire, lo que podr a influir
directamente en el fen meno que se est  modelando."

ggplot(data, aes(x = Dewpoint, y = Mean_temperature)) +
  geom_point()

"Las variables de presi n a nivel del agua tambi n tienen correlaci n con las
temperaturas. La presi n atmosf rica disminuye con el aumento de la elevaci n,
por lo que podr a intuirse que la temperatura media disminuye en lugares con
poca altitud en este conjunto de datos."

ggplot(data, aes(x = Mean_temperature, y = Sea_level_pressure)) +
  geom_point()

"Vamos a crear dos nuevas variables que resuman la informaci n de max_temperature,
min_temperature y las variables relativas a presi n."

data <- data %>%
  mutate(
    Diff_temperature = Max_temperature + Min_temperature,
    Diff_pressure = Standard_pressure - Sea_level_pressure
  ) %>%
  select(-Min_temperature, -Max_temperature, -Sea_level_pressure, -
    Standard_pressure)

"Vamos a realizar un an lisis de comprobaci n de outliers multivariantes, es
decir, en combinaci n con m ltiples variables. Primero, comprobamos
la normalidad con un test multivariante."

mvn(data = data, mvnTest = "hz")

"No lo pasa, por lo que los datos no siguen una normal con un alto grado de
confianza en altas dimensiones. Vamos a usar Mahalanobis igualmente para ver
la proporci n de potenciales outliers"

mahal_dist <- mahalanobis(data, colMeans(data), cov(data))
threshold <- qchisq(0.975, df = dim(data)[2])  # 97.5% confidence level

data <- as.data.frame(data) %>% mutate(Outlier = mahal_dist > threshold)

# % of potential outliers
(nrow(data %>% filter(Outlier)) / nrow(data) * 100)

ggplot(as.data.frame(mahal_dist), aes(x = mahal_dist)) +
```

```
  geom_histogram(
    binwidth = 0.5,
    fill = "lightblue",
    color = "black",
    alpha = 0.7
  ) +
  labs(title = "Mahalanobis distances distribution", x = "Mahalanobis
   Dist", y = "Freq") +
  theme_minimal()

"Tal y como se observa en el gr fico , la distribuci n de las
   distancias tampoco
sigue una normal. Es un indicativo m  s de que los datos no son
   normales. Por ello
no voy a eliminar las variables encontradas por este m todo."

"Por  ltimo  vamos a usar PCA para obtener un resumen de los datos y
   gr ficas que
poder analizar m  s que por reducir dimensionalidad, ya que solo
   tenemos 10
variables."

pca_res <- prcomp(data)
pca_df <- data.frame(pca_res$x)
ggplot(pca_df, aes(x = PC1, y = PC2)) +
  geom_point(color = "blue") +
  labs(title = "PCA - First Two Principal Components", x = "Principal
   Component 1", y = "Principal Component 2") +
  theme_minimal()

# Get the variance of each component and visualize it
explained_variance <- summary(pca_res)$importance[2, ]
cumulative_variance <- cumsum(explained_variance)

var_explained_df <- data.frame(
  Component = 1:length(explained_variance),
  Variance_Explained = explained_variance,
  Cumulative_Variance = cumulative_variance
)

# We visualize the variance explained by each component
ggplot(var_explained_df, aes(x = Component)) +
  geom_bar(
    aes(y = Variance_Explained),
    stat = "identity",
    fill = "blue",
    alpha = 0.7
  ) +
  geom_line(aes(y = Cumulative_Variance * max(Variance_Explained)),
            color = "red",
            size = 1) +
  labs(title = "Variance Explained by PCA Components", x = "Principal
   Component", y = "Variance Explained") +
```

```
  scale_y_continuous(sec.axis = sec_axis(~ . / max(var_explained_df$
    Variance_Explained), name = "Cumulative Variance")) +
  theme_minimal()

"Dada esta gr fica podemos observar que la varianza acumulada seg n se a aden
m s componentes deja de aumentar significativamente a partir de cuatro m s o menos.
De todas formas, con solo dos (para poder visualizarlo) ya se obtienen una varianza
alta y suficiente para obtener cierta informaci n."

# Biplot
autoplot(
  pca_res,
  data = data,
  colour = 'Mean_temperature',
  loadings = TRUE,
  loadings.label = TRUE,
  loadings.label.size = 3
)

"En este biplot se puede observar como temperaturas muy bajas se dan en lugares
con presiones altas y vientos altos, ya que los valores que se encuentran en
esa zona son combinaciones de valores altos para esas variables. Obviamente
valors bajos de las variables de temperatura (max, min) contribuyen much simo
a la temperatura media, pero por ser ta obvios son menos interesantes."
```

6.2. EDA Clasificación

```
library(tidyverse)
library(moments)
library("corrplot")
library(ggfortify)

"Leemos los datos saltandonos las cabeceras iniciales y despu s las parseamos
a mano. Seguido, vamos a ver unos pocos datos con head para hacernos una primera
idea."

data <- read.csv("newthyroid/newthyroid.dat",
                 skip = 10,
                 header = FALSE)
colnames(data) <- c(
  "T3resin",
  "Thyroxin",
  "Triiodothyronine",
```

```
  "Thyroidstimulating",
  "TSH_value",
  "Class"
)
head(data)

"Con la funci n de summary obtenemos un resumen estad stico general de cada
columna. Adem s vamos a obtener otros datos informativos como la dimensi n ,
y la estructura."

summary(data)
dim(data)
str(data)
colSums(is.na(data))

"Por lo que podemos ver, existen 6 variables o columnas (o caracter sticas) para
este conjunto. Existe gran variabilidad en caracter sticas como TSH_value y
Thyroidstimulating, donde valores muy bajos son cercanos al cero y muy altos
superan los 50.

Adem s, la mayor a de valores en estas variables parecen ser que rondan valores
bajos, esto nos lo puede indicar el tercer cuartil, que es en ambas variables
muy bajo en comparaci n al valor m ximo que toman.

Las variables son todas n mericas flotantes, excepto T3resin que es entera. No
hay valores faltantes en ninguna columna."

"Las clases a predecir en este problema de clasificaci n son las siguientes:"

table(data$Class)
prop.table(table(data$Class))

"Se puede observar un desbalanceo en el n mero de ejemplos para cada clase. La
clase 1 es muy mayoritaria."

"Vamos a normalizar todas las variables. Las que m s necesitan esta
transformaci n son las anteriormente mencionadas, que tienen valores muy
separados en el espacio. Las clases adem s, deben transformarse en factores
para poder tratarlas como categor as."

data <- data %>%
  mutate(Class = as.factor(Class)) %>%
```

```
  mutate(across(where(is.numeric), ~ (. - mean(., na.rm = TRUE)) / sd
    (.)))

selected_columns <- c("T3resin",
                      "Thyroxin",
                      "Triiodothyronine",
                      "Thyroidstimulating",
                      "TSH_value")

long_data <- data %>%
  pivot_longer(cols = selected_columns,
               names_to = "Variable",
               values_to = "Value")

ggplot(long_data, aes(x = Variable, y = Value, fill = Variable)) +
  geom_boxplot() +
  labs(title = "Boxplots", x = "Variable", y = "Value") +
  theme_minimal()

"Dada esta gr fica de boxplots podemos observar aquellos valores que
    se alejan
demasiado de la distribuci n de los datos. De hecho, el m todo de
    distancia
intercuartil es muy  til  para eliminar outliers."

q3 <- quantile(data$TSH_value, 0.75)
iqr <- IQR(data$TSH_value)
upper_bound <- q3 + 1.5 * iqr

# Porcentaje de "outliers" en TSH_value
length(data$TSH_value[data$TSH_value > upper_bound]) / length(data$TSH_
    value)

"Usando el m todo de distancia intercuartil (solo por arriba, ya que
    en el
boxplot podemos ver que no hay ninguno por abajo) encontramos
    potenciales
outliers. El porcentaje de ellos para la caracter stica seleccionada
    es muy
peque o. En el caso de que estuviese seguro (conocimiento de dominio)
    de que
esos valores son irreales, los quitar a, pero realmente no lo s  y el
quitarlos ahora me evitar a un posible futuro an lisis con el modelo
    de
predicci n usado, ya que a priori no s  si van a afectar severamente
    el
rendimiento de mi modelo."

"Creamos una peque a funci n para poder mostrar un gr fico de barras
    , de esta
forma podremos ver f cilmente las posibles distribuciones de cada
    variable."

plot_distribution <- function(data,
```

```
                                var_name,
                                binwidth = 0.1,
                                fill_color = "pink") {
  ggplot(data, aes_string(x = var_name)) +
    geom_density(fill = fill_color,
                 color = "black",
                 alpha = 0.7) +
    labs(
      title = paste("Distribution of", var_name),
      x = var_name,
      y = "Frequency"
    ) +
    theme_minimal()
}

plot_distribution(data, "TSH_value")

"Claramente tiene una asimetr a a la izquierda."

plot_distribution(data, "T3resin")

"T3resin tiene un aspecto que parece bastante normal. Ligeramente
   asim trica
hacia la derecha."

plot_distribution(data, "Thyroxin")

"Thyroxin tambi n tiene un aspecto que parece bastante normal."

plot_distribution(data, "Triiodothyronine")

"Triiodothyronine tiene una asimetr a a la izquierda."

plot_distribution(data, "Thyroidstimulating")

"Thyroidstimulating tiene una asimetr a a la izquierda muy acentuada."

"Dadas estas gr ficas de densidades de cada variable, podemos observar
    como se
distribuyen sus valores. Vemos que hay unas cuantas que cuentan con
   asimetr a
extrema. Estos que presentan esta asimetr a son aquellas variables que
anteriormente hab amos identificado con pocos valores extremos y
   muchos
valores bajos."

ggplot(long_data, aes(sample = Value)) +
  stat_qq() +
  stat_qq_line() +
  facet_wrap( ~ Variable) +
  labs(title = "Qqplot of every variable", y = "Variable quantiles", x
   = "Theoretical quantiles") +
  theme_minimal()

```

```
"T3resin y Thyroxin son las que se parecen m s a una normal, pero a n as tienen
variaciones en las colas. El resto de variables son muy asim tricas, ya se hab a
diagnosticado, pero es una confirmaci n m s.

Vamos a hacer un test de Shapiro sobre las dos variables m s prometedoras para
ver si podemos rechazar la hip tesis nula de que no siguen una distribuci n
normal."

shapiro.test(data$T3resin)
shapiro.test(data$Thyroxin)

"El test de shapiro es muy significativo para las dos variables m s
prometedoras, lo que indica que podemos rechazar las asumpciones de normalidad
que hemos estado construyendo por medio de las gr ficas. Ninguna variable de las
que tenemos sigue una distribuci n normal, esto lo tendremos en cuenta para la
fase de clasificaci n."

"Vamos a hacer test de aimetr a y de curtosis para las variables que sabemos
no son normales."

agostino.test(data$T3resin)
agostino.test(data$Thyroxin)
anscombe.test(data$T3resin)
anscombe.test(data$Thyroxin)

"Entendemos pues que los datos tienen asimetr a y curtosis, pues se rechazan en
ambos casos."


"Creamos otra funci n para ver la distribuci n, pero por clase. De esta forma
podemos observar que distribuciones de valores parecen asociarse a ciertas
clases."

plot_distribution_for_every_class <- function(data, var_name, binwidth = 0.6) {
  ggplot(data, aes(x = !!sym(var_name), fill = Class)) +
    geom_density(color = "black", alpha = 0.7) +
    labs(
      title = paste("Distribution of", var_name),
      x = var_name,
      y = "Frequency"
    ) +
    facet_wrap(~ Class) +
```

```
    theme_minimal() +
    theme(legend.position = "none")
}

plot_distribution_for_every_class(data, "Thyroidstimulating")
plot_distribution_for_every_class(data, "TSH_value")
plot_distribution_for_every_class(data, "T3resin")
plot_distribution_for_every_class(data, "Thyroxin")
plot_distribution_for_every_class(data, "Triiodothyronine")

"Parece observarse que para todas las variables, valores altos de la misma
suelen agruparse en la clase 1."

pca_res <- prcomp(data %>% select(-Class))

# Biplot
autoplot(
  pca_res,
  data = data,
  colour = 'Class',
  loadings = TRUE,
  loadings.label = TRUE,
  loadings.label.size = 3
)

"Vamos a mirar si existe alguna correlaci n entre variables, para ello podemos
usar dos m todos. Con pairs mostramos cada variable en relaci n a otra variable.
Lo bueno es que podemos de manera intuitiva identificar relaciones que vayan
m s all  de la linealidad. Con la librer a de correlaci n, podemos mostrar
gr ficamente junto al coeficiente las correlaciones entre variables."

numeric_data <- select(data, -Class)
pairs(numeric_data)

"Una de las relaciones m s notables parece estar entre TSH_value y
Thyroidstimulating, donde se observa una distribuci n bastante concentrada cerca
de los valores m s bajos, con algunos valores at picos.
Hay patrones interesantes entre T3resin y Thyroxin, que muestran una dispersi n
no aleatoria, sugiriendo alg n tipo de relaci n entre estas hormonas tiroideas.
Triiodothyronine tambi n muestra patrones de agrupaci n interesantes con otras
variables, especialmente visible en algunas de las dispersiones."

cor_matrix <- cor(numeric_data)
corrplot(cor_matrix,
         method = "number",
```

```
          type = "upper",
          tl.col = "black")

"En el caso de este dataset, no encuentro ninguna correlaci n demasiado grande a
excepci n de Triiodothyronine y Thyroxin, que presenten una correlaci n positiva
y algo lineal bastante interesante.
Dados estos resultados, yo no eliminar a ninguna por redundancia. Adem s , la
dimensi n del problema no es tal como para tener que optimizar el tama o
de las componentes / caracter sticas que tenemos, pues hay muy pocas. De hecho
muchas veces, dos caracter sticas muy correladas no implican redundancia. Un
ejemplo de esto aparece en Feature Extraction - Foundations and Applications
by I. Guyon et al. (p.10, figure 2 (e)), donde dos caracter sticas con
un ratio de correlaci n alto son totalmente necesarias (ambas) para la
clasificaci n del problema."

"En cuanto a transformaciones, es posible que fuese muy  til  en aquellas
aquellas variables con asimetr as largas a la derecha, en ese caso podr amos
aplicar transformaciones logar tmicas."

data <- data %>%
  mutate(
    log_Triiodothyronine = log1p(Triiodothyronine),
    log_Thyroidstimulating = log1p(Thyroidstimulating),
    log_TSH_value = log1p(TSH_value)
  )

plot_distribution(data, "log_Triiodothyronine")
plot_distribution(data, "Triiodothyronine")
plot_distribution(data, "log_Thyroidstimulating")
plot_distribution(data, "Thyroidstimulating")
plot_distribution(data, "log_TSH_value")
plot_distribution(data, "TSH_value")

shapiro.test(data$log_Triiodothyronine)
shapiro.test(data$Triiodothyronine)
agostino.test(data$log_Triiodothyronine)
agostino.test(data$Triiodothyronine)

shapiro.test(data$log_Thyroidstimulating)
shapiro.test(data$Thyroidstimulating)
agostino.test(data$log_Thyroidstimulating)
agostino.test(data$Thyroidstimulating)

shapiro.test(data$log_TSH_value)
shapiro.test(data$TSH_value)
```

```
agostino.test(data$log_TSH_value)
agostino.test(data$TSH_value)

"Se ha reducido la asimetr a notablemente, aunque eso no hace que las
distribuciones de las variables sean normales. "
```

6.3. Regresión

```
library(tidyverse)
library(kknn)
library(RWeka)

"Leemos los datos saltandonos las cabeceras iniciales y despu s las
   parseamos
a mano."

data <- read.csv("wankara/wankara.dat", skip = 14, header = FALSE)
colnames(data) <- c(
  "Max_temperature",
  "Min_temperature",
  "Dewpoint",
  "Precipitation",
  "Sea_level_pressure",
  "Standard_pressure",
  "Visibility",
  "Wind_speed",
  "Max_wind_speed",
  "Mean_temperature"
)
str(data)

"Vamos a aplicar las transformaciones realizadas en la parte del EDA."

data <- as.data.frame(scale(data))
data$Diff_temperature <- data$Max_temperature + data$Min_temperature
data$Diff_pressure <- data$Standard_pressure - data$Sea_level_pressure

data <- data[, !names(data) %in% c("Min_temperature",
                                   "Max_temperature",
                                   "Sea_level_pressure",
                                   "Standard_pressure")]

summary(data)
pairs(data)

"Ahora vamos a utilizar las 5 variables regresoras que m s sentido
   tengan seg n
nuestro criterio. Seg n lo visto en el EDA, en lo relativo a
   correlaci n y al
propio sentido com n, la variable m s prometedora es Diff_temperature
    (es la
```

```
combinaci n lineal de Max_temperature y Min_temperature). Seguido de Dewpoint,
Diff_pressure (combinaci n lineal de variables originales de presi n), Visibility
y Wind_speed.

Se ha tenido en cuenta las variables que parecen m s lineales con respecto a la
variable objetivo, o que al menos parezcan tener una relaci n en alg n sentido (por
ejemplo, Visibility es algo curva, algo logaritmica)."

perform_linear_regression <- function(data, target_var = "Mean_temperature", predictors) {
  regression_results <- list()

  for (predictor in predictors) {
    formula <- as.formula(paste(target_var, "~", predictor))

    fit <- lm(formula, data = data)

    plot_data <- data.frame(x = data[[predictor]],
                            y = data[[target_var]],
                            fitted = predict(fit))

    p <- ggplot(plot_data, aes(x = x, y = y)) +
      geom_point(alpha = 0.5) +
      geom_line(aes(y = fitted), color = "blue", size = 1) +
      labs(
        title = paste(target_var, "vs", predictor),
        x = predictor,
        y = target_var
      ) +
      theme_minimal()

    regression_results[[predictor]] <- list(model = fit,
                                            summary = summary(fit),
                                            plot = p)

    # Print summary and plot
    cat("\n--- Regression Results for", predictor, "---\n")
    print(regression_results[[predictor]]$summary)
    print(regression_results[[predictor]]$plot)
  }

  return(regression_results)
}

predictors <- c("Diff_temperature",
                "Dewpoint",
                "Diff_pressure",
                "Visibility",
                "Wind_speed")
regression_analysis <- perform_linear_regression(data, predictors =
```

```
    predictors)

"Claramente el mejor predictor de la temperatura media es diff_
    temperature, seguido
de Dewpoint. El RSE es extremadamente bajo usando ese predictor,
    adem s tiene un
R cuadrado ajustado del 0.98, lo que significa que con solo esa
    variable es posible
explicar el 98% de la variabilidad de la variable objetivo."

"Vamos a eliminar variables usando backwards selection."

fit1 <- data %>%
  lm(formula = Mean_temperature ~ ., data = .)

summary(fit1)

fit2 <- data %>%
  lm(formula = Mean_temperature ~ . - Max_wind_speed, data = .)

summary(fit2)

fit3 <- data %>%
  lm(formula = Mean_temperature ~ . - Max_wind_speed - Precipitation,
     data = .)

summary(fit3)

"A continuaci n vamos a ajustar un modelo de regresi n lineal
    m ltiple utilizando
varias variables y teniendo en cuenta relaciones no linales e
    interacciones. Aunque
los resultados son dif cilmente mejorables"

fit1 <- data %>%
  lm(formula = Mean_temperature ~ Diff_temperature, data = .)

summary(fit1)


fit2 <- data %>%
  lm(formula = Mean_temperature ~ Diff_temperature + Dewpoint + Diff_
    pressure,
     data = .)

summary(fit2)

fit3 <- data %>%
  lm(
    formula = Mean_temperature ~ Diff_temperature + Dewpoint + Diff_
    pressure + Visibility,
    data = .
  )

```

```
summary(fit3)

"A adir poco a poco todas las variables mejora el resultado, pero es
    m nimo. No creo
que sea un camino a seguir. Voy a probar con interacciones entre
    variables que
tengan sentido."

fit4 <- data %>%
  lm(formula = Mean_temperature ~ Diff_temperature * Dewpoint,
     data = .)

summary(fit4)

fit5 <- data %>%
  lm(formula = Mean_temperature ~ Diff_temperature * I(Dewpoint ^ 2),
     data = .)

summary(fit5)

fit6 <- data %>%
  lm(formula = Mean_temperature ~ Diff_temperature * I(Dewpoint ^ 2) *
    Dewpoint,
     data = .)

summary(fit6)

fit7 <- data %>%
  lm(
    formula = Mean_temperature ~ Diff_temperature * I(Dewpoint ^ 2) *
    Dewpoint * I(Diff_temperature ^

                   2),
    data = .
  )

summary(fit7)

"Parece que la interacci n de las dos variables m s importantes junto
    a no linealidad
mejora m s el resultado."

"Voy a probar con Visibilidad al cuadrado, ya que pareciese por la
    gr fica que
sigue una relaci n no lineal con respecto a la variable objetivo."

fit8 <- data %>%
  lm(formula = Mean_temperature ~ Diff_temperature + I(Visibility ^ 2),
     data = .)

summary(fit8)

"Probando varias combinaciones, lo que he podido ver es que el modelo
    parece
```

```
"ajustar mejor a adiendo interacciones entre varias variables en vez de solo
a adirlas. Alguna no linealidad (sobre todo con Dewpoint) parece a adir algo
de mejora. Voy a usar el modelo final del m todo backwards por la interpretabilidad
y porque el mejor modelo que usa interacciones y no linealidad (fit7) no mejora tanto
como para plantearse sacrificar interpretabilidad."

"Ahora vamos a aplicar regresi n con el algoritmo kNN."

fitknn1 <- kknn(Mean_temperature ~ ., data, data)

"Visualizamos los datos originales vs los puntos ajustados del knn"
ggplot(data = data, aes(x = Diff_temperature, y = Mean_temperature)) +
  geom_point() +
  geom_point(aes(y = fitknn1$fitted.values),
             color = "blue",
             shape = 20) +
  labs(x = "Diff_temperature", y = "Mean_temperature", title = "kNN") +
  theme_minimal()

"Calculamos el RMSE"
yprime <- fitknn1$fitted.values
sqrt(sum((data$Mean_temperature - yprime) ^ 2) / length(yprime))

"Utilizando todos los datos podemos obtener un RMSE pr cticamente igual que en
regresi n usando solo la variable de diff_temperature. Este problema concreto
parece bastante apto para una regresi n lineal, de todas formas probar
algunas combinaciones m s."

fitknn2 <- kknn(Mean_temperature ~ Diff_temperature, data, data)

"Calculamos el RMSE"
yprime <- fitknn2$fitted.values
sqrt(sum((data$Mean_temperature - yprime) ^ 2) / length(yprime))

"Usando solo la variable m s importante es capaz de conseguir un mejor ajuste
que la regresi n lineal."

ggplot(data = data, aes(x = Diff_temperature, y = Mean_temperature)) +
  geom_point() +
  geom_point(aes(y = fitknn2$fitted.values),
             color = "blue",
             shape = 20) +
  labs(x = "Diff_temperature", y = "Mean_temperature", title = "kNN") +
  theme_minimal()

"Podemos apreciar mucho menos ruido que usando todas las variables."
```

```
227
228 fitknn3 <- kknn(Mean_temperature ~ Diff_temperature + Dewpoint, data,
        data)
229
230 "Calculamos el RMSE"
231 yprime <- fitknn3$fitted.values
232 sqrt(sum((data$Mean_temperature - yprime) ^ 2) / length(yprime))
233
234 ggplot(data = data, aes(x = Diff_temperature, y = Mean_temperature)) +
235   geom_point() +
236   geom_point(aes(y = fitknn3$fitted.values),
237              color = "blue",
238              shape = 20) +
239   labs(x = "Diff_temperature", y = "Mean_temperature", title = "kNN") +
240   theme_minimal()
241
242 "A adiendo Dewpoint se consigue un ajuste con menor ruido incluso. Con
        diff_temperature
243 se pod a modelar una l nea muy buena que ajustase los datos, pero
        utilizando adem s
244 dewpoint parece que la varianza se modela mejor."
245
246 fitknn4 <- kknn(Mean_temperature ~ Diff_temperature * Dewpoint * I(
        Dewpoint ^
247                                                                    2)
        ,
248                 data,
249                 data)
250
251 "Calculamos el RMSE"
252 yprime <- fitknn4$fitted.values
253 sqrt(sum((data$Mean_temperature - yprime) ^ 2) / length(yprime))
254
255 ggplot(data = data, aes(x = Diff_temperature, y = Mean_temperature)) +
256   geom_point() +
257   geom_point(aes(y = fitknn4$fitted.values),
258              color = "blue",
259              shape = 20) +
260   labs(x = "Diff_temperature", y = "Mean_temperature", title = "kNN") +
261   theme_minimal()
262
263 "Si en vez de sumar las variables a adimos interacciones entre ellas y
        adem s tenemos
264 en cuenta un poco de no linealidad mejora el ajuste."
265
266 "He probado con otras combinaciones, pero no parece que encuentre nada
        mejor. Adem s
267 hay que tener en cuenta que quiz  estemos ajustando demasiado los
        resultados a la
268 muestra, lo cu l en un entrenamiento real llevar a al sobreajuste."
269
270 "Vamos a realizar 5-fold cross validation con el mejor modelo obtenido
        de regresi n
271 lineal m ltiple y a partir de ah , intentar obtener un mejor modelo
```

```
   con kNN. Despu s
compararemos todos los resultamos entre  s   y con el algoritmo M5."

run_k_fold_cv <- function(data, model = "lm", k = 5) {
  columns <- c(
    "Dewpoint",
    "Precipitation",
    "Max_temperature",
    "Min_temperature",
    "Sea_level_pressure",
    "Standard_pressure",
    "Visibility",
    "Wind_speed",
    "Max_wind_speed",
    "Mean_temperature"
  )

  colnames(data) <- columns

  train_mse_list <- numeric()
  test_mse_list <- numeric()

  data$Diff_temperature <- data$Max_temperature + data$Min_temperature
  data$Diff_pressure <- data$Standard_pressure - data$Sea_level_
   pressure

  data <- data[, !names(data) %in% c("Min_temperature",
                                     "Max_temperature",
                                     "Sea_level_pressure",
                                     "Standard_pressure")]
  idx <- sample(nrow(data))
  data <- data[idx, ]

  fold_size <- round(nrow(data) / k)

  results <- data.frame(
    Actual = numeric(),
    Predicted = numeric(),
    Fold = numeric(),
    MSE = numeric(),
    Train = numeric(),
    Test = numeric()
  )

  for (i in 0:(k - 1)) {
    start <- 1 + i * fold_size
    end <- min((i + 1) * fold_size, nrow(data))

    val <- data[start:end, ]
    train <- data[-(start:end), ]

    train_scaled <- scale(train)
    mean_train <- attr(train_scaled, "scaled:center")
    sd_train <- attr(train_scaled, "scaled:scale")
```

```
    train <- as.data.frame(train_scaled)

    val <- as.data.frame(scale(val, center = mean_train, scale = sd_
  train))

    if (model == "lm") {
      fit <- lm(Mean_temperature ~ ., data = train)
      train_pred <- predict(fit, newdata = train)
      yprime <- predict(fit, newdata = val)
    } else if (model == "kknn") {
      fit <- kknn(Mean_temperature ~ ., train = train, test = val)
      fit_train <- kknn(Mean_temperature ~ ., train = train, test =
  train)
      train_pred <- fit_train$fitted.values
      yprime <- fit$fitted.values
    } else if (model == "m5") {
      fit <- M5P(Mean_temperature ~ ., data = train)
      train_pred <- predict(fit, newdata = train)
      yprime <- predict(fit, newdata = val)
    }

    train_mse <- mean((train$Mean_temperature - train_pred) ^ 2)
    test_mse <- mean((val$Mean_temperature - yprime) ^ 2)

    train_mse_list <- c(train_mse_list, train_mse)
    test_mse_list <- c(test_mse_list, test_mse)

    results <- rbind(
      results,
      data.frame(
        Actual = val$Mean_temperature,
        Predicted = as.numeric(yprime),
        Fold = i + 1,
        MSE = test_mse,
        Train = tail(train_mse_list, 1),
        Test = tail(test_mse_list, 1)
      )
    )
  }

  results
}

data <- read.csv("wankara/wankara.dat", skip = 14, header = FALSE)
lm_results <- run_k_fold_cv(data, model = "lm")
knn_results <- run_k_fold_cv(data, model = "kknn")
m5_results <- run_k_fold_cv(data, model = "m5")

# Summary of results
lm_mse <- mean(lm_results$MSE)
knn_mse <- mean(knn_results$MSE)
m5_mse <- mean(m5_results$MSE)
```

```
lm_mse_train <- mean(lm_results$Train)
knn_mse_train <- mean(knn_results$Train)
m5_mse_train <- mean(m5_results$Train)

print(paste("Linear Regression MSE:", lm_mse))
print(paste("KNN MSE:", knn_mse))
print(paste("M5 MSE:", m5_mse))

plot_predictions <- function(results, model) {
  ggplot(results, aes(
    x = Actual,
    y = Predicted,
    color = as.factor(Fold)
  )) +
    geom_point(alpha = 0.7) +
    geom_abline(
      slope = 1,
      intercept = 0,
      linetype = "dashed",
      color = "black"
    ) +
    labs(
      title = paste(model, "-", "Results"),
      x = "Actual Mean Temperature",
      y = "Predicted Mean Temperature",
      color = "Fold"
    ) +
    theme_minimal()
}

plot_lm <- plot_predictions(lm_results, "Linear Regression")
plot_knn <- plot_predictions(knn_results, "KNN")
plot_m5 <- plot_predictions(m5_results, "M5")

print(plot_lm)
print(plot_knn)
print(plot_m5)

"Por los resultados de test y lo visto en las gr ficas , parece que el
   modelo que
mejor se ajusta a este dataset es regresi n lineal m ltiple seguido
   muy de cerca por knn."

"Vamos a comparar los algoritmos."

resultados_test <-  read.csv("regr_test_alumnos.csv")
resultados_train <- read.csv("regr_train_alumnos.csv")
tablatst <- cbind(resultados_test[, 2:dim(resultados_test)[2]])
colnames(tablatst) <- names(resultados_test)[2:dim(resultados_test)[2]]
rownames(tablatst) <- resultados_test[, 1]
tablatra <- cbind(resultados_train[, 2:dim(resultados_train)[2]])
colnames(tablatra) <- names(resultados_train)[2:dim(resultados_train)
   [2]]
rownames(tablatra) <- resultados_train[, 1]
```

```
427
428 tablatst[17, 1] <- lm_mse
429 tablatst[17, 2] <- knn_mse
430 tablatst[17, 3] <- m5_mse
431
432 tablatra[17, 1] <- lm_mse_train
433 tablatra[17, 2] <- knn_mse_train
434 tablatra[17, 3] <- m5_mse_train
435
436 "Normalizamos utilizando las diferencias relativas entre los resultados
      de los
437 algoritmos. Despu s , generamos una tabla con valores ajustados y
     procedemos al
438 test de Wilcoxon, que es un test por pares."
439
440 difs <- (tablatst[, 1] - tablatst[, 2]) / tablatst[, 1]
441 wilc_1_2 <- cbind(ifelse (difs < 0, abs(difs) + 0.1, 0 + 0.1),
442                   ifelse (difs > 0, abs(difs) + 0.1, 0 + 0.1))
443 colnames(wilc_1_2) <- c(colnames(tablatst)[1], colnames(tablatst)[2])
444 head(wilc_1_2)
445
446 LMvsKNNtst <- wilcox.test(wilc_1_2[, 1],
447                           wilc_1_2[, 2],
448                           alternative = "two.sided",
449                           paired = TRUE)
450 Rmas <- LMvsKNNtst$statistic
451 pvalue <- LMvsKNNtst$p.value
452 LMvsKNNtst <- wilcox.test(wilc_1_2[, 2],
453                           wilc_1_2[, 1],
454                           alternative = "two.sided",
455                           paired = TRUE)
456 Rmenos <- LMvsKNNtst$statistic
457 Rmenos
458 Rmas
459 pvalue
460
461 "Dado un p-valor de menos de 0.7, no se puede rechazar la hip tesis
     nula, por lo que
462 no podemos asegurar que existen diferencias estad sticamente
     significativas entre
463 KNN y LM."
464
465 "Ahora realizamos test m ltiples para comparar todos los algoritmos
     entre s  . Para
466 ello usamos el test de friedman."
467
468 test_friedman <- friedman.test(as.matrix(tablatst))
469 test_friedman
470
471 "Se aplica una correcci n para evitar errores acumulados."
472
473 tam <- dim(tablatst)
474 groups <- rep(1:tam[2], each = tam[1])
475 pairwise.wilcox.test(as.matrix(tablatst),
```

```
                      groups,
                      p.adjust = "holm",
                      paired = TRUE)

"Esto indica que el algoritmo 3 (M5) no tiene evidencia estad stica de ser mejor sobre
los otros dos algoritmos. En este test adem s se puede ver como el algoritmo 1 y el
2 (KNN vs LM) tampoco tienen diferencias significativas entre ellos, como se pudo
ver en el anterior an lisis."

"Vamos a hacer lo mismo con los resultados de train."

difs <- (tablatra[, 1] - tablatra[, 2]) / tablatra[, 1]
wilc_1_2 <- cbind(ifelse (difs < 0, abs(difs) + 0.1, 0 + 0.1),
                  ifelse (difs > 0, abs(difs) + 0.1, 0 + 0.1))
colnames(wilc_1_2) <- c(colnames(tablatra)[1], colnames(tablatra)[2])
head(wilc_1_2)

LMvsKNNtst <- wilcox.test(wilc_1_2[, 1],
                          wilc_1_2[, 2],
                          alternative = "two.sided",
                          paired = TRUE)
Rmas <- LMvsKNNtst$statistic
pvalue <- LMvsKNNtst$p.value
LMvsKNNtst <- wilcox.test(wilc_1_2[, 2],
                          wilc_1_2[, 1],
                          alternative = "two.sided",
                          paired = TRUE)
Rmenos <- LMvsKNNtst$statistic
Rmenos
Rmas
pvalue

sapply(tablatra, median)
sapply(tablatst, median)

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

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

"Lo esperado en este apartado era que se obtuvieran resultados m s significativos.
Si los algoritmos iban bien en test, en training deber an haber ido incluso mejor.
Aunque esta premisa no es cierta siempre, en este caso puede observarse como
```

```
los p-valores son m s bajos incluso. Por ello, parece que los mismos algoritmos siguen
siendo los mejores y no parece que se haya realizan un sobreajuste."
```

6.4. Clasificación

```
library(caret)
library(kknn)
library("MASS")
library(tidyverse)

"Leemos los datos saltandonos las cabeceras iniciales y despu s las parseamos
a mano. Seguido, vamos a ver unos pocos datos con head para hacernos una primera
idea."

data <- read.csv("newthyroid/newthyroid.dat",
                 skip = 10,
                 header = FALSE)
colnames(data) <- c(
  "T3resin",
  "Thyroxin",
  "Triiodothyronine",
  "Thyroidstimulating",
  "TSH_value",
  "Class"
)
summary(data)

"Normalizamos los datos usando el escalado z-score. Esto hace que los datos sigan
una distribuci n con media 0 y desviaci n t pica 1. Esto es muy necesario cuando
se tienen datos en distintas escalas, como es el caso, y se utilizan algoritmos
que usan distancias."

data <- data %>% mutate(Class = as.factor(Class))
class_column = data$Class
data <- as.data.frame(scale(data[1:ncol(data) - 1]))
data$Class <- class_column

data <- data %>%
  mutate(
    Thyroidstimulating = log1p(Thyroidstimulating),
    TSH_value = log1p(TSH_value)
  )
summary(data)

"Vamos a dividir el conjunto d datos en conjuntos de train y test. Para eso definimos
```

```
la siguiente funci n"

train_test_split <- function(data,
                             test_percentage = 0.2,
                             seed = 42) {
  set.seed(seed)
  data <- data[sample(1:nrow(data)), ]

  n <- round((1 - test_percentage) * nrow(data))

  train <- data[1:n, 1:ncol(data)]
  test <- data[(n + 1):nrow(data), 1:ncol(data)]

  list(train = train, test = test)
}

results <- train_test_split(data)
train <- results$train
test <- results$test
str(results)

"Una vez divididos los conjuntos podemos empezar a experimentar con KNN
   ."
fit1 <- kknn(Class ~ .,
             train = train,
             test = test,
             k = 1)

make_confusion_matrix <- function(knn_model) {
  predicted <- fitted(knn_model)
  conf_matrix <- table(Predicted = predicted, Actual = test$Class)
  conf_matrix_df <- as.data.frame(as.table(conf_matrix))

  plot <- ggplot(conf_matrix_df, aes(x = Actual, y = Predicted, fill =
   Freq)) +
    geom_tile() +
    geom_text(aes(label = sprintf("%d", Freq)),
              color = "white",
              size = 10) +
    scale_fill_gradient(low = "cyan", high = "darkgreen") +
    theme_minimal() +
    theme(
      axis.title = element_text(size = 12, face = "bold"),
      axis.text = element_text(size = 10),
      legend.title = element_text(size = 10),
      plot.title = element_text(
        size = 14,
        face = "bold",
        hjust = 0.5
      )
    ) +
    labs(
      title = "Confusion Matrix",
      x = "Actual Class",
```

```
      y = "Predicted Class",
      fill = "Count"
    ) +
    coord_equal()

  accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
  print(plot)
  print(paste("Accuracy:", round(accuracy, 2)))
}

make_confusion_matrix(fit1)

"Dadas estas m tricas parece que el knn est  generalizando muy bien con k=1, lo
cual es bastante asombroso, pues k=1 es un valor muy dado al sobreajuste ya que
el modelo se hace mucho m s sensible a cambios cuando a adimos nuevos datos, se
hace muy variable."

"Vamos a probar varios valores de k para ver como var an los resultados."
fit2 <- kknn(Class ~ .,
             train = train,
             test = test,
             k = 5)

make_confusion_matrix(fit2)


fit3 <- kknn(Class ~ .,
             train = train,
             test = test,
             k = 15)

make_confusion_matrix(fit3)


fit4 <- kknn(Class ~ .,
             train = train,
             test = test,
             k = 40)

make_confusion_matrix(fit4)

fit5 <- kknn(Class ~ .,
             train = train,
             test = test,
             k = 100)

make_confusion_matrix(fit5)

"El modelo con k cada vez mayor pierde precisi n, porque al hacer esto, el
```

```
modelo se suaviza y se vuelve m s general. Esto puede reducir el sobreajuste
al hacer que las predicciones dependan de un n mero mayor de vecinos, lo que
ayuda a promediar los datos y a reducir la sensibilidad a las fluctuaciones
peque as. Sin embargo, si kk es demasiado grande, el modelo comienza a perder
capacidad para captar patrones espec ficos, lo que puede llevar a un subajuste
(underfitting), ya que los puntos de datos m s cercanos pueden no tener tanta
influencia sobre la predicci n final. "

"En mi caso escoger a k=5. Aunque k=1 sea mejor y me haya dado mejores resultados
en test, a adir un poco de regularizaci n en este caso, considero que no deval a
demasiado la calidad del modelo y previene de posibles ajustes demasiado finos. Si
bien es verdad que el resultado de test es mejor para k=1 y por tanto el modelo,
incluso con k=1 parece no sobreajustar nada en datos nunca vistos, prefiero ser
precavido."

"Ahora vamos a usar LDA para ajustar un modelo al conjunto de datos. Pero primero
han de comprobarse ciertas asunciones"

"Los datos son normales para cada clase."
str(data)
test_variable <- function(var_name) {
  resultados <- data %>%
    group_by(Class) %>%
    summarize(
      p_value = if (is.numeric(.data[[var_name]]) &&
                    all(!is.na(.data[[var_name]]))) {
        shapiro.test(.data[[var_name]])$p.value
      } else {
        NA
      },
      normal = ifelse(p_value > 0.05, "Puede ser normal", "No es normal"),
      .groups = "drop"
    )
  plot <- ggplot(data, aes(sample = .data[[var_name]])) +
    stat_qq() + stat_qq_line() +
    facet_wrap(~ Class)
  print(plot)
  print(resultados)
}

test_variable("T3resin")
```

```
test_variable("Thyroxin")
test_variable("Triiodothyronine")
test_variable("Thyroidstimulating")
test_variable("TSH_value")

"Parece que el test de Shapiro no rechaza para algunas variables dentro de algunas
clases. Si bien no se puede decir que no son normales, tampoco se pueden rechazar.
Dada esta evidencia y la de los qqplots, no se verifica la primera asunci n de LDA."

"Para comprobar si cada clase tiene matrices de varianze-covarianza id nticas, se
puede utilizar el test de Bartlett"

bartlett.test(T3resin ~ Class, data)
bartlett.test(Thyroxin ~ Class, data)
bartlett.test(Triiodothyronine ~ Class, data)
bartlett.test(Thyroidstimulating ~ Class, data)
bartlett.test(TSH_value ~ Class, data)

"Se rechaza el test de Bartlett para todas las variables, lo que significa que
las varianzas de los grupos o muestras comparadas no son iguales, es decir, no
se cumple el supuesto de homogeneidad de varianzas (homocedasticidad)."

"Clasificamos con LDA"
lda_model <- lda(Class ~ ., data = train)
lda_model
lda.pred.train <- predict(lda_model, train)
lda.pred.test <- predict(lda_model, test)

plot_data <- lda.pred.train$x %>%
  as_tibble() %>%
  mutate(Class = train$Class)

"Mostramos el gr fico donde se muestra como se distribuyen las observaciones de
las clases en el espacio generado por LDA utilizando las primeras dos componentes
lineales discriminantes (LD1 y LD2)."

"Se puede observar que las clases est n bien separadas en el espacio generado por
LDA. Una buena separaci n indica que el modelo ha logrado discriminar
correctamente entre clases."

ggplot(data = plot_data) +
  geom_point(aes(x = LD1, y = LD2, color = Class)) +
  scale_colour_manual(
    name = "Class",
    values = c("red", "green", "blue"),
```

```
    labels = c("1", "2", "3")
  ) +
  labs(title = "Data Transformed After LDA")

t <- table(lda.pred.test$class, test$Class)
t

sum(diag(t)) / nrow(test)

plot_data <- lda.pred.test$x %>%
  as_tibble() %>%
  mutate(known = test$Class, # Replace with the correct column for class labels
         prediction = lda.pred.test$class) %>%
  pivot_longer(c("prediction", "known"),
               names_to = "Type",
               values_to = "Class")

ggplot(data = plot_data) +
  geom_point(aes(
    x = LD1,
    y = LD2,
    shape = Type,
    color = Class
  )) +
  scale_colour_manual(
    name = "Species",
    values = c("red", "green", "blue"),
    labels = c("1", "2", "3")
  ) +
  scale_shape_manual(name = "Type", values = c(5, 3)) +
  labs(title = "Validation Data + Predictions Transformed After LDA")

"Pese a no cumplirse las normalidades de datos por clase y la igualdad de matrices
de covarianza-varianza, el modelo es capaz de generalizar muy bien y de encontrar
separabilidad entre las clases."

qda_model <- qda(Class ~ ., data = train)
qda_model
qda.pred.train <- predict(qda_model, train)
qda.pred.test <- predict(qda_model, test)

t <- table(Predicted = qda.pred.test$class, Actual = test$Class)
t

sum(diag(t)) / nrow(test)

conf_matrix_df <- as.data.frame(as.table(t))

ggplot(conf_matrix_df, aes(x = Actual, y = Predicted, fill = Freq)) +
  geom_tile() +
  geom_text(aes(label = sprintf("%d", Freq)), color = "white", size =
```

```
    10) +
  scale_fill_gradient(low = "cyan", high = "darkgreen") +
  theme_minimal() +
  theme(
    axis.title = element_text(size = 12, face = "bold"),
    axis.text = element_text(size = 10),
    legend.title = element_text(size = 10),
    plot.title = element_text(size = 14, face = "bold", hjust = 0.5)
  ) +
  labs(title = "Confusion Matrix",
       x = "Actual Class",
       y = "Predicted Class",
       fill = "Count") +
  coord_equal()

"Vamos a comparar los tres algoritmos."

k_fold_cross_validation <- function(data,
                                    k = 10,
                                    model_function,
                                    metric_function,
                                    seed = 42) {
  set.seed(seed)
  folds <- sample(1:k, size = nrow(data), replace = TRUE)
  performance_metrics <- c()

  for (i in 1:k) {
    test_indices <- which(folds == i)
    train_indices <- setdiff(1:nrow(data), test_indices)

    train_data <- data[train_indices, ]
    test_data <- data[test_indices, ]

    model <- model_function(train_data, test_data)
    if (inherits(model, "kknn")) {
      predictions <- fitted(model)
    } else {
      predictions <- predict(model, test_data)
    }

    if (inherits(model, "lda") | inherits(model, "qda")) {
      predictions_class <- predictions$class
    } else {
      predictions_class <- predictions
    }

    performance <- metric_function(predictions_class, test_data$Class)
    performance_metrics <- c(performance_metrics, performance)
  }

  mean_performance <- mean(performance_metrics)
  std_deviation <- sd(performance_metrics)

  return(list(mean = mean_performance, std_deviation = std_deviation))
```

```
}

model_function_knn <- function(train_data, test_data) {
  model <- kknn(Class ~ .,
                train = train_data,
                test = test_data,
                k = 5)
  return(model)
}

model_function_lda <- function(train_data, test_data) {
  model <- lda(Class ~ ., data = train_data)
  return(model)
}

model_function_qda <- function(train_data, test_data) {
  model <- qda(Class ~ ., data = train_data)
  return(model)
}

metric_function_accuracy <- function(predictions, actual) {
  confusion_matrix <- table(predicted = predictions, actual = actual)
  accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
  return(accuracy)
}

result_knn <- k_fold_cross_validation(
  data,
  k = 10,
  model_function = model_function_knn,
  metric_function = metric_function_accuracy
)
print(paste(
  "KNN Accuracy: ",
  round(result_knn$mean, 4),
  "  ",
  round(result_knn$std_deviation, 4)
))

result_lda <- k_fold_cross_validation(
  data,
  k = 10,
  model_function = model_function_lda,
  metric_function = metric_function_accuracy
)
print(paste(
  "LDA Accuracy: ",
  round(result_lda$mean, 4),
  "  ",
  round(result_lda$std_deviation, 4)
))

result_qda <- k_fold_cross_validation(
  data,
```

```
  k = 10,
  model_function = model_function_qda,
  metric_function = metric_function_accuracy
)
print(paste(
  "QDA Accuracy: ",
  round(result_qda$mean, 4),
  "  ",
  round(result_qda$std_deviation, 4)
))
```

7. Bibliografía

[1] Y. Shao y M. Zhou, "A characterization of multivariate normality through univariate projections," *Journal of Multivariate Analysis*, vol. 101, n.º 10, págs. 2637-2640, 2010, ISSN: 0047-259X. DOI: https://doi.org/10.1016/j.jmva.2010.04.015. URL: https://www.sciencedirect.com/science/article/pii/S0047259X10001168.

[2] I. Guyon, S. Gunn, M. Nikravesh y L. Zadeh, *Feature extraction. Foundations and applications. Papers from NIPS 2003 workshop on feature extraction, Whistler, BC, Canada, December 11–13, 2003. With CD-ROM*. ene. de 2006, vol. 207, ISBN: 978-3-540-35487-1. DOI: 10.1007/978-3-540-35488-8.