Facultad de Ciencias

Métodos de detección de anomalías y clustering en series temporales
(Methods for anomaly detection and clustering in time series)

Trabajo de Fin de Máster
para acceder al

MÁSTER EN CIENCIA DE DATOS

Autor: Alonso del Saso, Javier

Director\es: Van Vaerenbergh, Steven Johan Maria

Septiembre - 2020

Índice

1. Agradecimientos

Antes de nada, me gustaría expresar mi más sinceros sentimientos de gratitud hacia mi Director del proyecto, Steven Johan Maria Van Vaerenbergh por su constante apoyo en el proyecto además de su inmensa paciencia y conocimientos. Sus continuos esfuerzos y guías me han permitido escribir esta tesis.

Aparte de mi Director, quiero dar gracias a mi Codirector Carlos Meneses Agudo por la motivación que me dio para investigar y desarrollar este proyecto además de muchas ideas que permitieron mejorar la calidad de esta investigación.

También quiero dar las gracias a Lara Lloret por su continuo apoyo durante el curso académico y reconocerle el esfuerzo y mérito como profesora y científica.

A todos mis compañeros del Máster por su excelente forma de colaborar, ofrecer ayuda durante el curso y crear un gran ambiente de trabajo para todos.

Por último pero no menos importante, quiero agradecer a todos los miembros de CIC por ofrecerme la oportunidad de colaborar con ellos y conocer las dinámicas del trabajo en equipo.

2. Resumen

El análisis de series temporales es un campo fundamental para el procesamiento de la información. En el mundo real, el contexto de los datos a menudo se desconoce y el experimentador debe realizar un esfuerzo por extraer información limitada. En este proyecto se estudia un conjunto de algoritmos no supervisados para el análisis de series temporales correspondientes al consumo de agua de una zona residencial. En particular, los algoritmos se centran en detección de anomalías, reducción de la dimensionalidad y clustering de los datos. En una serie de experimentos, se comparan los resultados obtenidos por los diferentes algoritmos de cada categoría.

Abstract

Time series analysis is a fundamental field for information processing. In the real world, the context of the data is often unknown and the experimenter must make an effort to extract limited information. In this project, we study a set of unsupervised algorithms for the analysis of time series corresponding to the water consumption of a residential area. In particular, the algorithms focus on anomaly detection, dimensionality reduction and data clustering. In a series of experiments, the results obtained by the different algorithms in each category are compared.

Palabras clave: clasificación, series temporales, medidas de distancia, representaciones, detección de anomalías, aprendizaje no supervisado

Keywords: *clustering, time-series, distance measure, representations, anomaly detection, unsupervised learning*

3. Introducción

La investigación y empeño del estudio de series temporales en el mundo tecnológico levanta un enorme interés en desarrollar métodos y técnicas eficientes para su análisis. Durante los últimos 10 años, se han recopilado muchos algoritmos y métodos distintos para aplicarse en variables que cambian con el tiempo. Mucho de los fines relacionados con estos estudios están relacionados con el mundo de finanzas (e.g. detección de fraudes), mundo médico, para el estudio de la climatología y meteorología [1, 6]. Mucha de la investigación se centra en desarrollar algoritmos eficientes y eficaces.

Los análisis de las series temporales pueden clasificarse en dos grupos según la información de los datos, pueden ser supervisados (cuando conocemos la etiqueta real de una serie, e.g. la serie de temperatura en una región cuando se produce una tormenta) y no supervisados (no podemos poner en contexto los resultados de los análisis). En el mundo real existen muchas bases de datos sin etiquetar debido al coste que supondría (proceso manual, lento y con posibles errores humanos).

4. Fundamento teórico y estado del arte

A continuación, profundizaremos en las bases de los algoritmos empleados en la detección de anomalías y clasificación para conjuntos grandes de datos. Existen muchas metodologías distintas dependiendo de la finalidad con que se analizan los datos.

Las aplicaciones de la detección de anomalías y clasificación son muchas y varían según el campo de conocimientos en el que se emplean. Por ejemplo, se pueden detectar irregularidades en las transacciones que realiza una empresa con sus clientes. En el mundo de la automatización y robótica, la detección de anomalías en las lecturas y diagnósticos de máquinas permiten evadir obstáculos sin costes muy elevados o daños.

La detección de anomalías se puede realizar de los siguientes modos:

Detección supervisada, cuando se conoce las clases anómalas y normales. Permite entrenar modelos de predicción. Sin embargo, presenta dos problemas fundamentales, la precisión de los datos etiquetados como anomalías, suelen ser difíciles de clasificar, y por otro lado, datos considerados anómalos son menos comunes y produce dificultades para ser encontrados por modelos de entrenamiento y predicción.

Detección semisupervisada, cuando se conoce qué datos se encuentran dentro de la normalidad, generalmente gracias a la ayuda de modelos del problema. Esto permite encontrar anomalías que se alejen de los resultados del modelo.

Detección no supervisada, se desconoce la naturaleza de los datos. Se define formalmente a una anomalía a partir de definiciones de similitud y distancia entre los datos. Es la forma más utilizada en el mundo práctico por sus diversas aplicaciones. Esta es la forma que usamos en este proyecto.

4.1. Series temporales

Para proceder con el proyecto, establecemos algunas definiciones esenciales sobre las series temporales, su interés en el mundo práctico y algunas aplicaciones.

Una serie temporal es una secuencia ordenada de datos cuyas variables son continuas (en el caso de ser discretas, se conoce como secuencia temporal) y las variables cambian en función del tiempo [7]. Los datos suelen componerse de varias variables y su tamaño es, en algunos casos, considerable y a tener en cuenta para el costo computacional [18].

Entre los métodos desarrollados para el estudio de series temporales [1] se encuentran

Detección dinámica de cambios en la serie temporal, en especial de correlaciones entre series e.g. valores en los mercados de acciones de una empresa que guarden una correlación con las acciones de otra.

Predicción y recomendación en base a métodos de clustering y funciones de aproximación para dichos clústers. Por ejemplo, modelos de predicción de meteorología en una región en base

a temperatura, presión atmosférica, etc.

Descubrimiento de patrones, por ejemplo, en las ventas de productos en una tienda que permitan maximizar los beneficios.

Detección de anomalías, patrones fuera de la norma de los datos. En este proyecto aplicamos algoritmos que usen esta metodología para la detección de irregularidades en el consumo de agua.

Para aplicaciones de agrupamientos y clasificación [1, 13], las series temporales se pueden organizar en:

- **Series temporales completas** esto es, cuando las series temporales son discretas e individuales. Esta es la forma más común de atacar problemas relacionados con series temporales.
- **Series temporales secuenciadas**. Las series están subdivididas en secuencias generadas por una ventana deslizante sobre la misma. Esta forma de organizar los datos ha sido cuestionada por la comunidad, los resultados de los algoritmos son esencialmente aleatorios [13].
- **Puntos temporales**. A cada dato de la serie se le asigna las distancias a los puntos más próximos además de una medida de similitud con los mismos.

4.2. Anomalía

Existen varias clases de anomalías [6] según se clasifique con el resto de los datos.

En primer lugar tenemos las *anomalías puntuales*. Cuando un dato individual se considera que se encuentra fuera de la norma, se dice que este dato es una *anomalía puntual*. Son el tipo de anomalía más estudiado puesto que su definición no es específica de la naturaleza de los datos o del problema en cuestión. Para este proyecto consideraremos algoritmos que permitan detectar estar anomalías.

Por otro lugar, están las *anomalías contextuales*. Cuando se encuentran datos que son anómalos según el contexto pero podría no serlo si los comparamos con todo el conjunto de datos. Para la detección de dichas anomalías es necesario conocer la estructura de los datos previo a su estudio.

Las *anomalías colectivas* se generan cuando un conjunto de los datos se determina que es anómalo con respecto al resto. No es necesario que cada dato individual sea una anomalía.

4.3. Medidas de similitud

Para la detección de anomalías, es necesario establecer una medida que cuantifique la similitud (i.e. la distancia) entre dos series temporales.

Consideremos $\mathbf{x} = (x_1, \cdots, x_n)$ y $\mathbf{y} = (y_1, \cdots, y_m)$ series temporales cuyos elementos se encuentran representados en un espacio N-dimensional continuo y arbitrario $x_i, y_j \in \chi^N,\ 1 \leq i \leq n$ y $1 \leq j \leq m$. Consideremos una función ψ que mida la discrepancia o similitud entre dos puntos de cada serie. En el caso más común se aplica la distancia cuadrática, $\psi(x_i, y_j) = ||x_i - y_j||^2$.

4.3.1. Distancia euclídea

Se define a partir de la función de discrepancia ejemplar que se propuso anteriormente. Para dos series $\mathbf{x},\ \mathbf{y}$, la distancia euclídea D_{eucl} viene dada como:

$$D_{eucl}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{i=n} \psi(x_i, y_i)} = \sqrt{\sum_{i=1}^{i=n} ||x_i - y_i||^2} \tag{1}$$

Es la más eficiente en cuanto a coste. Notemos que es la distancia es definida positiva. Esta medida padece de dos problemas fundamentales, ambas series deben tener el mismo tamaño para computarse su distancia. Por otro lado, es muy ineficiente en recoger las características (o *features*) de las series, en particular si ambas tienen las mismas formas pero están distorsionadas en el tiempo.

Por estas razones, no se suele aplicar esta distancia en aplicaciones reales de manejo de series temporales.

4.3.2. DTW

La DTW o *Distance Time Warping* surgió como propuesta para medir mejor las distancias que la euclídea. Se definen dos vectores indexados $\pi_x,\ \pi_y$ cuya longitud $p_x,\ p_y \leq n + m - 1$ tal que $\pi_x(1) = 1 \leq \cdots \leq \pi_x(p) = n$ y $\pi_y(1) = 1 \leq \cdots \leq \pi_y(p) = m$. Estos dos vectores indican un recorrido por la matriz W, cuyo elemento $\mathrm{W}(i, j)$ se define como:

$$\mathrm{W}(i, j) = \psi(x_i, y_j) = ||x_i - y_j||^2 \tag{2}$$

La DTW es aquella que minimiza la distancia desde $\mathrm{W}(1, 1)$ hasta $\mathrm{W}(n, m)$:

$$D_{dtw}(\mathbf{x}, \mathbf{y}) \overset{min}{=} \sqrt{\sum_{i=1}^{p} \mathrm{W}(\pi_x(i), \pi_y(i))} \tag{3}$$

Además, para asegurar la continuidad y monotonicidad del recorrido por la matriz W decimos que $0 \leq \pi_x(i+1) - \pi_x(i) \leq 1$ y $0 \leq \pi_y(j+1) - \pi_y(j) \leq 1$.

Esta nueva forma de definir distancias permite distorsionar y escoger qué elemento de una serie se empareja con qué elemento de la otra. De esta forma, la distancia se minimiza cuando

se encuentra las mismas *features* en ambas series aunque no aparezcan en el mismo instante de tiempo. También se consigue que la distancia se pueda computar aún cuando no tienen la misma longitud.

Este método resulta relativamente costoso y se suelen aplicar algoritmos que limitan la forma del recorrido, como la banda de Sakoe-Chiba o el paralelogramo de Itakura ambos basándose en que el recorrido mínimo debe encontrarse cerca de la diagonal principal de la matriz W [14].

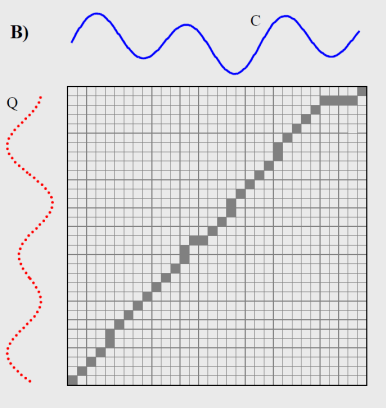


Figura 1: Cada cuadrado representa un elemento de la matriz W, el recorrido está marcando las casillas que minimizan la distancia. Cada columna corresponde a un dato de la serie C y cada fila a la serie Q. Observamos que si el recorrido fuera la diagonal, estaríamos en el caso de la distancia euclídea. Imagen extraída de [14]

4.3.3. GA Kernel

El Global Alignment Kernel (GA Kernel) es una aproximación más novedosa de obtener medidas de similitud entre series. Consiste en contabilizar todas las posibles distancias exponenciales. Definimos una función divergente que es la exponencial de la función de similitud $\kappa = e^{-\psi}$. Consideramos $\mathcal{A}(n, m)$ como el conjunto de todas las posibles trayectorias de la matriz W para obtener una medida de similitud, es decir, $\pi \in \mathcal{A}(n, m)$.

El kernel de alineamiento global es la suma de todas las posibles trayectorias:

$$D_{k_{GA}}(\mathbf{x},\ \mathbf{y}) = \sum_{\pi \in \mathcal{A}} e^{-D_{\mathbf{x},\mathbf{y}}(\pi)} = \sum_{\pi \in \mathcal{A}} \prod_{i=1}^{p} \kappa(x_{\pi_x(i)},\ y_{\pi_y(i)}) \tag{4}$$

La función κ suele definirse como el kernel gaussiano parametrizado por un parámetro λ que evita que priorize la diagonal de la matriz W. También se consigue un kernel combinando el kernel gaussiano con el triangular y se han observado resultados similares [8].

El resultado recoge más características que la DTW puesto que esta tiene en cuenta no sólo la trayectoria óptima, sino todas las posibles.

4.4. Representación de las series temporales

Métodos de representación (o métodos de reducción de la dimensionalidad) son muy útiles con las series temporales. Primero, se reduce el espacio que ocupan los datos en memoria. Además, el cálculo de distancias se agiliza considerablemente haciendo factible métodos de clustering sobre las series temporales.

Un método de representación es un mapeado de la serie $\mathbf{x} = (x_1, \cdots, x_n)$ a $\mathbf{x}' = (x'_1, \cdots, x'_{n'})$ donde $n' < n$. El objetivo de un método de reducción de la dimensión es simplificar la dimensionalidad manteniendo la estructura y correlación entre los datos, esto es, dos series que son similares en el espacio original deben serlo en el espacio reducido.

4.4.1. Shapelets

Los *shapelets* son subsecuecias temporales máximas discriminantes. Secciones de las series temporales que mejor agrupan y clasifican a las secuencias. Aporta información sobre las clases además de las características en cada grupo de series y consigo, una interpretación de los datos.

Definimos la distancia $M_{i,k}$ como la distancia mínima entre el shapelet S_k con alguna subsecuencia en la serie $\mathbf{x}_i$:

$$M_{i,j} = \min_{j=1\cdots L} \frac{1}{L} \sum_{l=1}^{L} (\mathbf{x}_{i,j+l-1} - S_{k,l})^2 \tag{5}$$

Se puede construir una matriz que reúna la afinidad de cada serie con cada *shapelet*. Se reduce la dimensionalidad del problema al número de *features* o *shapelets*. Se combina con una función de pérdida de clasificación mediante un hiperplano para capturar la clase a la que pertenece cada serie. Este método permite la predicción de futuras series correspondientes a cada clase.

Uno de los mayores problemas que encontramos con este método es el coste computacional, con optimizaciones, el coste es lineal con el número de series pero cuadrático con su tamaño [10]. Además, el algoritmo tiene 6 hiperparámetros que debe optimizar y aumenta el coste.

4.4.2. PCA

El método de componentes principales [12] es uno de los más populares cuando se trata de reducir la dimensionalidad de los datos. Su limitación más notable es la detección de la linealidad global de los datos. Sin embargo, la proyección de los datos a un espacio sin correlaciones lineales permitiría a otros métodos de reducción detectar estructuras más complejas.

4.4.3. LLE

Local Linear Embedding (LLE) busca estructuras no lineales en los datos que se pueden extraer partiendo de ajustes lineales locales. Se considera que los datos se encuentran embebidos en una hipersuperficie suave y se puede estimar variaciones lineales con distancias cortas.

El algoritmo busca reconstruir cada punto a partir de una combinación lineal de los puntos cercanos. La función de coste viene dada como la diferencia cuadrática de cada representación con el mismo dato:

$$\epsilon(W) = \sum_i \left|\left| \mathbf{x}_i - \sum_j W_{ij}\mathbf{x}_j \right|\right|^2 \tag{6}$$

Donde W_{ij} es el peso del dato $\mathbf{x}_j$ para reconstruir $\mathbf{x}_i$. Se considera el sumatorio a los $j-$primeros vecinos de $\mathbf{x}_i$. Para encontrar la representación reducida de los datos se propone una función de coste como en (6) con vectores de dimensión $d < D$.

El algoritmo trabaja bien para cualquier número de dimensiones. Una de sus principales limitaciones es el coste que presenta resolver un problema matricial de $k \times k$ para encontrar los autovectores de cada dato. Además, podemos observar que el mapeado tiende a unir distintas clases lo que complica su interpretación [20, 24].

4.4.4. Isomapas

Los isomapas (o mapeados característicos isométricos) recogen las distancias entre un par de datos, como el recorrido más corto a través de los datos. La distancia será por tanto, la suma de las distancias entre pares de puntos conectados por el camino. Se puede entender como la reconstrucción aproximada de una geodésica en un espacio (propio de los datos) embebido dentro de un espacio mayor (i.e. el espacio en el que están representados).

Se construye una matriz de distancias D_{ij} formadas por geodésicas, se extraen los primeros p-autovectores y p-autovalores que formarán la nueva representación de los datos [23].

Este algoritmo se fundamenta en el espacio donde se encuentran los datos, por lo que permite extraer estructuras no lineales a gran escala.

Encontrar los vectores del espacio embebido puede resultar el proceso más costoso. Aunque los resultados son favorables [23]. Otros algoritmos más avanzados son capaces de conseguir resultados más fiables aunque con mayor costes [24].

4.4.5. t-SNE

El algoritmo *t-distributed Stochastic Neighbor Embedding* (t-SNE) mapea cada distancia entre par de series por una probabilidad condicionada que funciona como medida de similitud. La probabilidad $P(\mathbf{x}_i|\mathbf{x}_j)$ de que la serie $\mathbf{x}_i$ sea vecina de x_j es mayor cuanto más próxima se encuentre. En el espacio reducido, los datos deben mantener el mismo grado de similitud $Q(\mathbf{x}'_i|\mathbf{x}'_j)$.

Se aplica la divergencia de Kullback-Leibler para determinar el grado de similitud que existe entre el espacio original y el reducido. El coste computacional es cuadrático con el número de datos.

Este método evita la acumulación de puntos en el centro además de que revela estructuras a distintas escalas con un mismo mapeado [24].

Algunas de las limitaciones que presenta este algoritmo es su coste, (e.g. PCA es más rápido). Como la divergencia de Kullback-Leibler no es convexa, diferentes inicializaciones del algoritmo pueden resultar en distintos mapeados. Además, la estructura global no está expresamente recogida en el método (al contrario que en PCA).

4.5. Detección de anomalías

A continuación, se ofrece un listado de algunos de los algoritmos y métodos no supervisados empleados para la detección de anomalías.

4.5.1. BDSCAN

Density Based Spatial Clustering of Applications with Noise (DBSCAN) permite encontrar datos anómalos y de forma simultánea definir grupos o clústers de datos [9].

DBSCAN comienza por destacar que los clústers se encuentran en regiones con alta densidad de puntos mientras que zonas de baja densidad son regiones que no pertenecen a ningún clúster. Esta idea intuitiva del agrupamiento de datos permite que el algoritmo se adapte a la distribución de datos en el espacio sin imponer condiciones a su estructura.

El algoritmo asigna con una etiqueta a los datos denominados "*muestras nucleares*" como aquellas que dentro de un radio contienen un número mínimo de vecinos, $N(p)_\epsilon \geq MinPts$ donde p es el punto en cuestión y $N(p)_\epsilon$ es el número de puntos que se encuentran dentro de un radio ϵ y MinPts es el número de vecinos mínimo para considerar p como una muestra nuclear.

Los datos que no pasen el umbral pero se encuentran a distancia ϵ de una muestra nuclear (i.e. *directly density-reachable*, *ddr*) se denominan muestras fronterizas. El conjunto de las muestras fronterizas que son *ddr* a una misma nube de muestras nucleares conforman un clúster.

Aquellos datos que no cumplen las condiciones para ninguno de los dos tipos de puntos, se denominan anomalías. En otras palabras, puntos en zonas de baja densidad que no pertenecen a ningún clúster son anómalos.

DBSCAN funciona particularmente bien con puntos de alta dimensionalidad debido a su eficiente coste ($\mathcal{O}(n\ \log\ n)$). Además, es independiente de las estructuras que forman los datos.

Hoy en día sigue siendo un método a destacar por su aplicación en diversos campos con resultados comparables a otros algoritmos más avanzados [21].

4.5.2. OPTICS

Ordering Points to Identify the Clustering Structure (OPTICS) es un método que se puede entender como una generalización de DBSCAN. OPTICS asigna a cada dato un orden y valor de alcance. El segundo se calcula como la distancia más pequeña para que el punto en cuestión

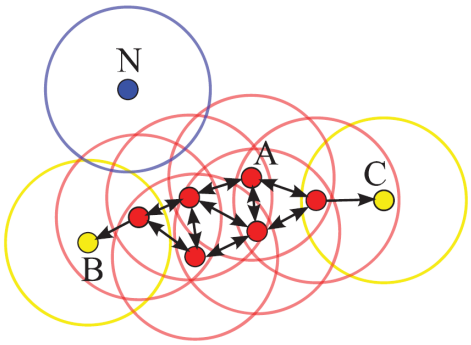


Figura 2: Los puntos rojos son muestras nucleares (minPts = 4), los amarillos son fronterizos y el azul es un punto anómalo. Los radios ϵ se representan por un anillo alrededor de cada punto. Extraido de [21].

sea *ddr* a una muestra nuclear. El primero se obtiene ordenando los puntos según su proximidad a sus vecinos más cercanos.

Este algoritmo produce una representación que permite visualizar los distintos agrupamientos que se producen en los datos indiferente de la dimensionalidad. Otra ventaja que incluye el algoritmo es la detección de clústers con distinta escalabilidad. Este era uno de los limitantes de DBSCAN, existen agrupamientos con densidades muy altas que pueden formar otros grupos con menor densidad. Esto se debe gracias a la variación del radio ϵ.

Para datos de muy altas dimensiones la distancia euclídea puede no ser suficiente o correcta para ser empleada. Además, el algoritmo puede ser exhaustivo y ineficiente para la computación.

4.5.3. LOF

Local Outlier Factor (LOF) es un algoritmo que asigna un valor numérico a cada punto que representa su grado de anomalía. Aquellos puntos que se encuentren en zonas altamente pobladas tendrán un factor menor que aquellos en zonas de baja densidad.

La definición de anomalía que propone LOF parte de las distancias de los vecinos locales. De esta manera, es una anomalía aquellos datos que se encuentren en zonas de baja densidad comparadas con vecinos. Esto permite la detección de anomalías alrededor de clústers de densidad variable.

Para determinar si un dato entra en la categoría anómala, se decide de un límite en el factor de LOF donde los datos con valores superiores al establecido se consideran outliers. La forma de discretizar las anomalías se deja a manos del científico y no existen generalizaciones para aplicar.

4.6. Clustering de series temporales

A continuación se listan algunos de los métodos empleados en series temporales para clustering.

4.6.1. K-means

El método de K-means es uno de los más sencillos para la clasificación no supervisada [1, 6]. Emplea un método muy eficaz por lo que se aprovecha en datos masivos.

4.6.2. Modelos ocultos de Markov

También conocido como HMM (*Hidden Markov Models*)[4] busca el conjunto de estados $\mathcal{Y}$ ocultos y junto a los parámetros θ, generan los datos observados (i.e. el conjunto incompleto de datos) $\mathcal{X}$.

Se define una distribución de probabilidad para el conjunto completo $\mathcal{Z} = (\mathcal{X}, \mathcal{Y})$, $P(\mathcal{X}, \mathcal{Y}|\theta)$. Incorporamos la función de máxima verosimilitud para esta distribución.

Para encontrar estados ocultos, se define la distribución incompleta $p(\S|\theta)$ como una composición lineal de distribuciones (generalmente gaussianas) de tal forma que los estados ocultos y_i indiquen qué distribuciones componen qué variable x_i.

Este análisis estadístico del problema revela estructuras subyacentes a los datos. Para encontrar una clasificación no hace falta más que observar cuál es el estado oculto que mejor representa a x_i.

4.6.3. BIRCH

BIRCH (*Balanced Iterative Reducing and Clustering using Hierarchies*) es un método diseñado para la clasificación de grandes cantidades de datos.

A cada clúster propuesto, se le asigna un vector de valores que lo caracteriza, el $\mathbf{C}F$ (i.e., *Clustering Factor*). Luego, se construye un árbol de clústers de forma jerárquica tal que las hojas previas representen clústers de mayor tamaño y que su división produzca nuevos nodos (clústers) de menor tamaño.

El procedimiento finaliza una vez que los clústers alcanzan un tamaño menor que un parámetro dado o se obtenga el número de clústers deseado.

Este es un método de aprendizaje online además de eficiente puesto que para estudiar un clúster sólo hace falta prestar atención a su *CF* [26].

4.7. Métodos de puntuación

A la hora de comparar distintos algoritmos, es necesario obtener un método de comparación objetivo. Propondremos dos índices distintos para agrupaciones en sistemas de aprendizaje no supervisados.

Tanto para la comparativa de algoritmos de clustering como para algoritmos de detección de anomalías emplearemos los mismos índices. En la detección de anomalías, existen dos clases (datos normales y anormales) aunque para los métodos de clustering la puntuación sea más

alta al considerar mejores agrupaciones de datos, podemos seguir comparando los métodos de detección entre ellos.

4.7.1. Silhouettes

Este algoritmo puntúa la disimilitud de un dato i con su clúster asignado (i.e. la distancia media con los datos del mismo clúster, $a(i)$) comparado con la disimilitud de datos del clúster más cercano $b(i)$ [19].

Para un dato i se asigna un valor $s(i)$ de la forma:

$$s(i) = \frac{b(i) - a(i)}{\mathrm{m}ax(a(i),\ b(i))} \tag{7}$$

El rango de $s(i)$ es $[-1,\ 1]$. Para $s(i) = -1$, la disimilitud es máxima para datos del mismo clúster; para $s(i) = 0$ no existe diferenciación entre un clúster u otro; para $s(i) = 1$, la disimilitud es máxima para el clúster más cercano y mínimo para el propio, este valor resulta de aplicar un algoritmo de clasificación ideal. Para obtener un único valor para el algoritmo, aplicamos la media a todos los $s(i)$.

Este índice es uno de los más usados en la comunidad por su simpleza en el uso exclusivo de distancias. Es versátil a la hora de comparar distintos métodos de clustering sin conocer la etiqueta real de cada dato.

4.7.2. Calinski

También se conoce como el criterio del ratio de varianza (VCR). El índice de Calinski proporciona una relación entre el tamaño de un clúster y su distancia a otros [5]. Si c_q es el punto central del clúster C_q y c_E es el punto medio de todo el conjunto de datos, B_k es la dispersión inter-grupal y W_k es la dispersión intra-grupal:

$$\begin{aligned} B_k &= \sum_q n_q ||c_q - c_E||^2 \\ W_k &= \sum_q \sum_{x \in C_q} ||x - c_q||^2 \end{aligned} \tag{8}$$

Luego la medida de Calinski vendrá dada como:

$$s = \frac{\frac{B_k}{k-1}}{\frac{W_k}{n-k}} \tag{9}$$

Donde n y k son el número de datos y de clústers respectivamente.

Este método ofrece un rango de $s > 0$. Cuanto mayor es la puntuación, los clústers están mejor separados.

Ambos índices funcionan con la noción típica de un clúster. Esto puede suponer valoraciones erróneas dependiendo del método de clustering. Por ejemplo, si en un espacio $d = 2$ se encuentra que un clúster tiene forma de anillo que encierra a otro clúster. Ambos clústers compartirían el mismo centro aunque las distancias entre datos del mismo grupo serían muy distintas.

5. Metodología

Previos trabajos se han dedicado a clasificar y organizar una amplia variedad de métodos analíticos de datos temporales [1, 6] según el origen de los datos o el tipo de base que se emplea al construir un algoritmo, entre otros.

Este proyecto está enfocado en comprender, aplicar y analizar distintas metodologías de reducción de dimensión, detección de anomalías y clustering en series temporales de datos sin conocimiento previo (i.e. no supervisado). Los métodos serán contrastados y seleccionados para el análisis.

Para la evaluación de los métodos se empleará índices que cuantifiquen la calidad de clasificar de los métodos. En el último paso, también contrastaremos visualmente los datos para comprobar que la base de empleo de estos índices sea robusta.

Los datos empleados en este proyecto fueron ofrecidos por Consulting Informático de Cantabria (CIC) en el desempeño de conocer y emplear nuevas técnicas de análisis. Los datos, de una compañía de distribución de agua, proceden del consumo en una pequeña región a las afueras de Madrid durante 5 años.

Para el análisis, se ha empleado un ordenador Toshiba L-1607035 con procesador Intel(R) Core(TM) i7-6500U de 2,50 GHz y RAM de 8,00 GB con un sistema operativo Windows 10 Pro (versión: 2004, 64 bits). Se ha hecho uso de *Jupyter Notebooks* para el desarrollo del código.

El repositorio de este proyecto se encuentra en GitHub [3]. Los datos son de libre acceso y se pueden encontrar en Zenodo [2].

6. Desarrollo

A continuación describiremos la base con la que se desarrolla el análisis de los datos.

Python es el lenguaje de preferencia para proceder con el tratamiento de los datos. Permite escribir un código flexible y legible al mismo tiempo que compacto. Muchos de los algoritmos y métodos empleados ya se encuentran en librería que se emplean en este proyecto [11, 15–17, 22, 25].

6.1. Registro de datos

Los datos entregados vienen en formato `csv` y recogen las variables de la presión (kg/cm^2), el caudal (m^3/h) y el volumen total de agua consumida hasta ese momento (m^3) con una

frecuencia de 15 minutos. Existen variables adicionales que guardan los valores diarios de las 3 anteriores.

Cada variable está almacenada en un fichero `csv`. Los ficheros contienen rangos de fechas, el nombre codificado de la variable (e.g. la presión tiene asignado el nombre `GES450100001N00005`) y el uso horario i.e., la hora a la que fue tomada cada dato. Los registros se corresponden con una línea donde aparece la fecha y hora de la lectura, el valor numérico y un código representado por un entero. Esto último será ignorado puesto que no podemos interpretarlo y no aporta información.

Comenzamos por reagrupar las variables principales en otro archivo para mayor facilidad a la hora de proceder con su análisis.

6.2. Mantenimiento de datos

En (cleaning.ipynb) extraemos las variables y las reorganizamos en `VPQ.csv`. Incluimos las librerías de Python `numpy`, `pandas` y `matplotlib` para realizar cálculos, organizar en tablas y tomar visualizaciones pre-análisis respectivamente.

`numpy` es una de las herramientas básicas para realizar operaciones simples sobre una gran cantidad de datos, además de que es una dependencia a la librería de `Pandas`; ambas nos permitirán organizar en tablas de fácil acceso todos los datos, además de que `Pandas` nos aporta muchas funciones para estructurar los datos como creamos conveniente. Utilizaremos `matplotlib` para visualizar mejor qué valores toman las variables y si son coherentes.

A partir del archivo `metadata.xlsx`, interpretamos el nombre de cada variable y realizamos una lectura de cada uno de los archivos `csv`. Obtenemos una tabla con cada columna representando a cada variable y en cada fila, cada uno de los registros identificados por la fecha y hora que fueron tomados.

Encontramos que en los datos aparecen incongruencias con las horas de registro. Más en detalle, faltan los datos durante el cambio de hora del mes de octubre en los años comprendidos entre 2016 y 2019. Inclusive a los cambios de hora, existen saltos de tiempo de longitud variable (fig. 3).

Aplicamos un algoritmo de interpolación para eliminar los cambios de hora. En los espacios de tiempo en que se adelanta una hora, consideramos que la variación de los registros es lineal y generamos los datos para esa hora. Asimismo, en los cambios a una hora menos, se toma el valor medio.

6.3. Procesamiento de los datos

En esta sección se explican los pasos dados para ayudar a la interpretación de los datos una vez limpiados y de los algoritmos que se han usado para procesarlos (processing.ipynb).

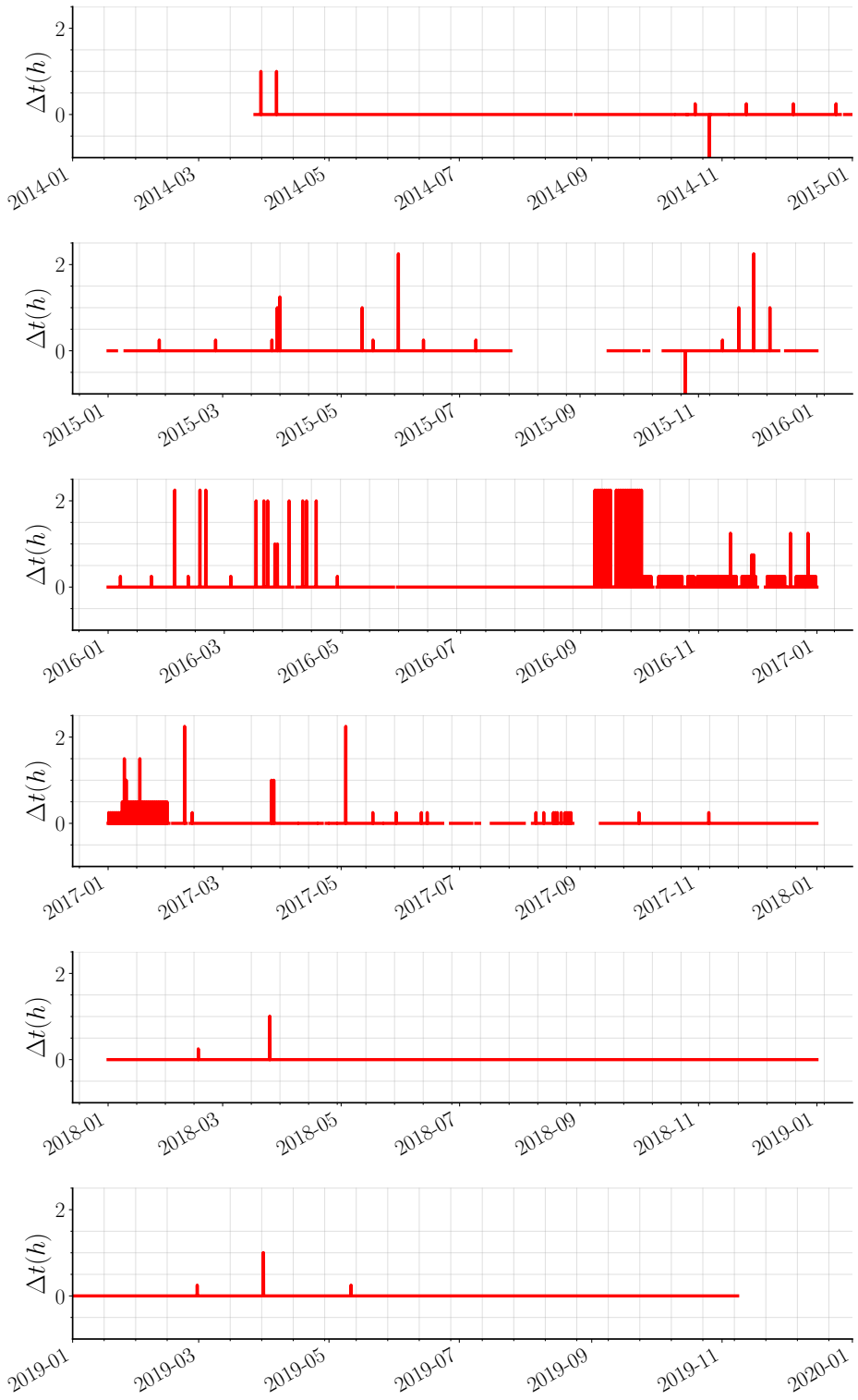
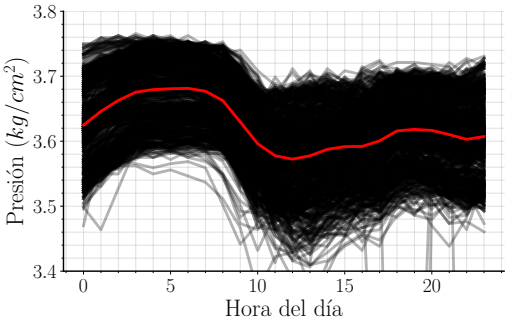


Figura 3: Representación de la difrencia de tiempo entre un registro y otro que sea contiguo. Los datos en el eje horizontal representan los saltos normales (de 15 minutos), las líneas en vertical indican que no hay datos registrados a esas horas. Cuanto mayor la línea vertical, mayor es el salto de tiempo. Los saltos mayores de 3 horas no se representan aunque podemos visualizarlos consultando el eje horizontal y comprobar que no hay datos.

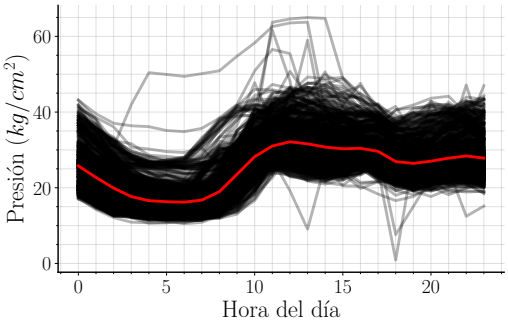
6.3.1. Secuencia de la serie

La serie original consiste en un listado de fechas que recogen las variables de la presión y el caudal según el consumo de agua de una zona residencial. Secuenciamos la serie en espacios de 24 horas, un registro por cada hora del día. De esta forma, pasamos de un espacio unidimensional para cada variable a uno 24-dimensional ($\mathbf{x} \in \mathbb{R}^{N\times 1} \rightarrow \mathbf{x}' \in \mathbb{R}^{N'\times 24}$).

Se toma como premisa que las diferencias de consumo de agua entre un día y otro para una misma hora son pequeñas comparadas con las variaciones a lo largo de cada ciclo. El proceso es cíclico con una frecuencia de 24 horas. Las variaciones y tendencias fuera de los ciclos son mucho menores que los cambios dentro de cada ciclo (fig. 4). De esta forma podemos encontrar comportamientos periódicos al mismo tiempo, tendencias globales que afecten a conjuntos de ciclos.



(a) Secuencias de 24 horas para la presión.



(b) Secuencias de 24 para el caudal.

Figura 4: Observamos que los ciclos diarios son similares entre ellos. Al mismo tiempo, cada secuencia es única y esto produce un ensanchamiento en el eje vertical de la forma principal

6.3.2. Estructura de los datos

Los datos se encuentran en una forma vectorial, en concreto podemos entender que los datos están en un espacio $\mathbb{R}^{d\times v}$ donde d es la dimensionalidad del vector ($d = 24$) y v es el número de variables ($v = 2$, i.e. presión y caudal).

Podemos representar la dependencia entre las variables con una figura, esto puede revelar estructuras aunque de forma limitada. Del mismo modo, tomamos la distancia de cada serie $\mathbf{x}_i$ a la serie media, la forma de la distribución de distancias puede revelar agrupaciones que se encuentren a una cierta distancia del "*centro*". Para encontrar relaciones entre agrupaciones de datos debemos

La reducción de la dimensionalidad del problema permite visualizar relaciones entre los datos más específicas además de darnos una mejor visión global de estas relaciones.

Antes de aplicar métodos de reducción de la dimensión es necesario realizar unos pasos previo. Re-escalamos las series para que puedan ser lo más homogéneas entre ellas ($\mu = 0, \sigma = 1$).

Muchos de los algoritmos y métodos que aplicaremos no puede trabajar con entradas matriciales. Mapeamos las variables a un espacio de $d' = d + v$ i.e. unimos los vectores correspondientes a cada variable por separado en un sólo vector $d = 48$. Esta transformación será la misma para cada algoritmo por lo que se puede entender cada dato como un registro de todos los valores de presión y caudal de un día.

6.3.3. Detección manual de anomalías

Los algoritmos de reducción y de clustering funcionan mejor cuando los datos no presentan ningún tipo de anomalía. Como trabajo preliminar, a partir de visualizar los datos en el espacio original escogemos aquellos que se encuentren muy alejados de la norma.

6.3.4. Matriz de distancias

Evaluamos las distancias entre los puntos para las distintas métricas (euclídea, DTW y GAK). Aunque este proceso sea muy costoso y llegue a tardar en el orden de minutos, se agiliza mucho los algoritmos que se basan en las distancias.

Construimos una matriz de distancias M^q tal que $M_{ij} =< \mathbf{x}_i, \mathbf{x}_j >^q$ donde q es la métrica empleada.

Obtendremos dos matrices por cada métrica, una por cada variable. Realizamos una combinación de las distancias para simplificar la matriz M^q y los resultados de algoritmos que las emplean.

$$\begin{aligned} D^P &= \sum_i (d_i^P)^2 \\ D^Q &= \sum_i (d_i^Q)^2 \end{aligned} \tag{10}$$

Las 3 métricas se basan en la noción de la distancia euclídea (la suma de los cuadrados). Si D^P es la suma de distancias para la variable presión en alguna métrica y D_Q es la correspondiente para el caudal 10, tomamos la suma para una nueva distancia $D = D^P + D^Q$. Para la distancia DTW, la suma implica que estamos forzando al camino pasar por $W(n^P, m^P)$ y el siguiente valor $W(n^P + 1, m^P + 1)$ para hallar la distancia óptima (recordar la forma de W, 2).

Para aplicar esta transformación podemos combinar las matrices de distancias que se generan a partir de cada variable. En el caso euclídeo y DTW, la distancia es simplemente la suma de los cuadrados (ec. 11). Para la métrica con el GAK, es necesario aplicar una transformación previa. Este método da una medida de similitud. Para poder comparar distancias, aplicamos $D^q = -\log(K^q)$ (recordemos la forma de K, 4). Luego para hallar una distancia, simplemente se suman (ec. 12).

$$D^{eucl} = \sqrt{(D^{eucl,P})^2 + (D^{eucl,Q})^2} \tag{11}$$

$$D = D^P + D^Q \tag{12}$$

6.3.5. Reducción de la dimensionalidad

Para reducir $\mathbf{x}_i \in \mathbb{R}^{d'}$ a un espacio representable $\mathbf{x}'_i \in \mathbb{R}^D$, se emplean los algoritmos de shapelets, PCA, LLE, isomapas y t-SNE. Buscamos reducir la dimensión a $D = 3$ antes que $D = 2$ puesto que la primera puede ofrecer más información. La posibilidad de elegir una dimensionalidad muy baja puede provocar colapsos donde encontremos densidades altas de puntos sin un equivalente en el espacio original.

La comparativa entre los métodos será puramente visual. Su funcionalidad es presentar estructuras en los datos para poder entenderlas. Muchos de los algoritmos obtienen resultados cuyos parámetros o mapeado no es comparable a otro (e.g. t-SNE compara la similitud desde un punto de vista estadístico mientras que LLE realiza aproximaciones en las distancias). Mucho de los esfuerzos en la comunidad científica terminan en un test visual del algoritmo para comprobar su eficacia [20, 23, 24].

6.3.6. Detección de anomalías

Para detectar *outliers* aplicamos los algoritmos DBSCAN, OPTICS y LOF. La optimización de los hiperparámetros de cada uno de ellos será elegido en base a los índices Silhouette y Calinski.

Los índices son muy distintos y para compararlos debemos detallar mejor el proceso que seguiremos con ellos. Ambas puntuaciones tratan de evaluar una posible agrupación de los datos. En esta sección nos centramos en detectar anomalías (primera agrupación), el resto de datos se consideran normales (segunda agrupación). Una limitación de estas medidas es el hecho de entender una agrupación como conjunto convexo por lo que las puntuaciones no reflejarán una *buena* evaluación de la clasificación pero suficiente para comparar entre los métodos.

Combinamos el índice Silhouette (s) con el Calinski (c) de tal forma que tengamos un único valor para establecer el mejor método. Comenzamos por normalizar por separado cada puntuación, para encontrar los valores en los mismos rangos (i.e. mismos pesos). Sumamos ambos calificadores aplicando un factor f de "*importancia*" que tiene cada calificador 13.

$$\text{calf} := f \cdot s_N + (1 - f) \cdot c_N \tag{13}$$

Los métodos de BDSCAN y LOF detectan las anomalías directamente. OPTICS, asigna un valor de *reachability* a cada punto. Cuanto mayor sea el valor, más alejado estará de sus vecinos. En ese caso, variamos un umbral de alcance según cuantiles de los datos. Asignamos el alcance óptimo al que maximice la puntuación en 13.

6.3.7. Clustering de las series temporales

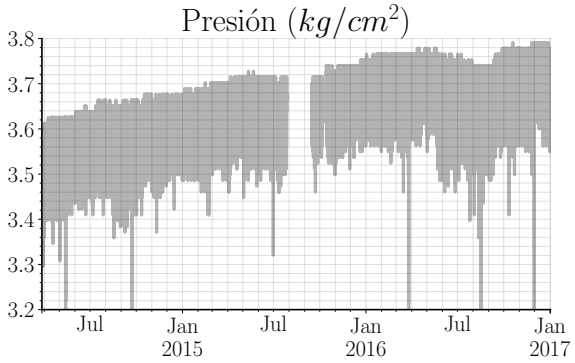
Aplicamos HMM, K-means y Birch a los datos. En algunos de los métodos se realizará una búsqueda automática de hiperparámetros dependiendo de lo difícil que resulte encontrar buenas agrupaciones.

Con HMM, escogemos distribuciones gaussianas para definir los estados ocultos. Podemos tomar la secuencia media de cada estado oculto y con una representación reducida observar dónde cae con respecto al resto de datos del mismo clúster.

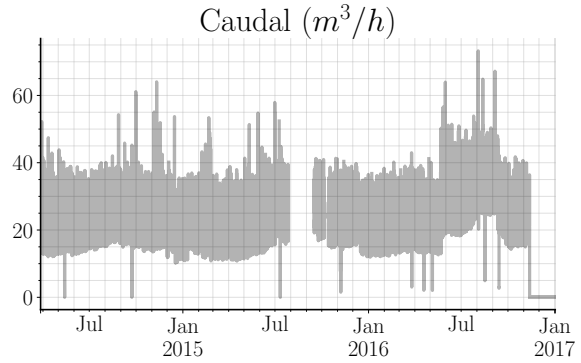
Con K-means y Birch, estudiaremos las puntuaciones en función del número de clústers. El algoritmo de K-means está desarrollado en `scikit`. Desafortunadamente, para encontrar los centroides, este método aplica la métrica euclídea o la DTW pero no está desarrollada para aceptar matrices de distancias o la métrica del GAK.

6.4. Método experimental

Una vez aplicado cada algoritmo a los datos, compararlo y observar sus resultados. Compilamos un método que combina cada una de las categorías. El objetivo es conseguir una visión más clara de las estructuras embebida en los datos eliminando datos anómalos. La clasificación obtenida se representará en un espacio reducido para visualizar y obtener las conclusiones del análisis.



(a) Secuencia de datos para la presión. La región no coloreada (julio de 2015) significa que no hay datos válidos para esas fechas. Observamos una tendencia creciente de 3,5 a 3,7 que se estabiliza en marzo de 2016.



(b) Secuencias de datos para el caudal. Faltan datos en el mes de agosto de 2015 y a partir de noviembre de 2016. No se observan tendencias claras.

Figura 5: Secuencias completas de las series de presión y caudal una vez acotada las fechas y arreglado los saltos de hora. Encontramos saltos en las series (incompletas).

7. Resultados y discusiones

Presentamos las series completas iniciales para tomar una idea preconcebida de los rangos y variaciones de cada variable. Observamos en (fig 5) saltos en los datos además de caídas

repentinas. La variación diaria de ambas variables es considerable respecto a la variación en la serie completa. Esto justifica reorganizar los datos en vectores diarios.

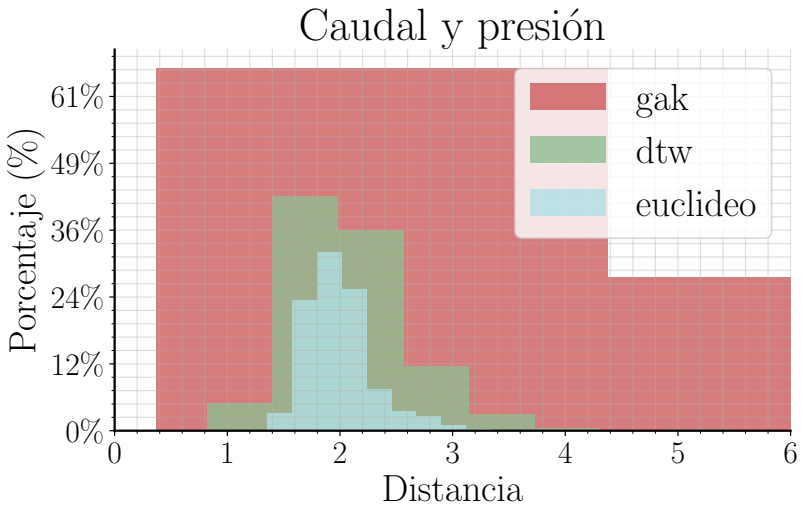


Figura 6: Distribución de distancias para las variables combinadas de presión y caudal. La altura representa el porcentaje de datos que caen en cada caja. Aparecen 3 distribuciones, una por cada métrica aplicada. Encontramos que las 3 tienen una forma similar y que además guardan la misma media.

Medimos la serie media, hayamos el valor medio aritmético de todas las series. Aparece representado en la figura 4. En la figura 6 evaluamos la distribución de distancias entre cada dato y la serie media. Podemos observar que la curva sigue una forma parecida a la de una distribución de χ^2 para las 3 métricas aplicadas. Encontramos que las formas de las distribuciones son muy parecidas, tienen los mismos centros y formas similares. La diferencia principal es la anchura de la distribuciones, siendo la métrica euclídea la que tiene una menor varianza y la GAK la que más.

Es interesante determinar que la forma proviene de χ^2. Esto indica que la distribución de cada dato sigue una forma parecida a la gaussiana cuyo parámetro μ es la serie media. También podemos concluir que la dispersión de los datos es mayor en la matriz de distancias del GAK y esto tendrá un impacto en los resultados como veremos más adelante.

En la figura 7 vemos la estructura de los datos originales. Parece existir una pequeña correlación lineal entre la presión y el caudal. Observamos un contorno definido en la región de la izquierda y una cola en la zona inferior derecha. Las mismas zonas corresponden a las horas de madrugada (alrededor de las $4:00$) y del mediodía respectivamente. Podemos observar también 3 regiones de alta concentración de puntos, cuando el caudal está por debajo de $20m^3/h$.

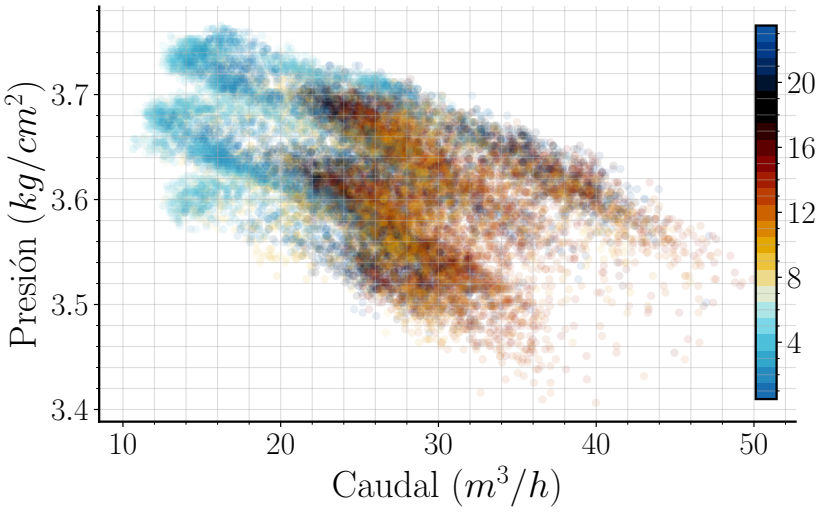


Figura 7: Representación de la presión frente al caudal. El código de colores representa las horas del día. Los tonos anaranjados corresponden al mediodía y los azulados a medianoche.

Los datos son parte de una estructura embebida de $d = 24$ dimensiones. Para llegar a visualizar mejor estas agrupaciones nos apoyamos en métodos de reducción de la dimensión.

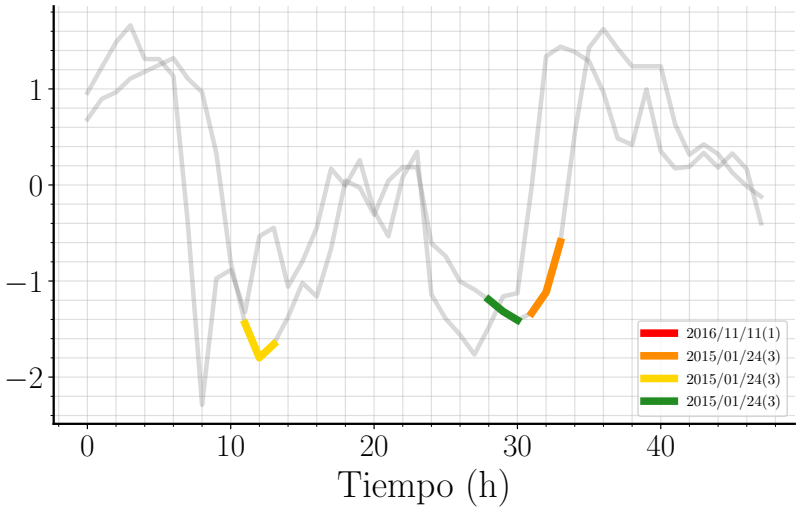


Figura 8: En código de colores son shapelets, en gris son las series que contienen dichos shapelets para la combinación de las variables de presión y caudal. Las primeras 24 horas corresponden a la presión mientras que las otras 24 corresponden al caudal. La leyenda indica la fecha de cada serie además del número de componentes del shapelet (entre paréntesis).

7.1. Algoritmos de reducción

7.1.1. Shapelets

Aplicamos el algoritmo de shapelets determinando que hay como máximo 10 subseries que extraer. Expondremos los shapelets que mejor determinan al sistema y de las series a las que pertenecen.

En (8) observamos los 4 mejor shapelets obtenidos. Encontramos que son extremadamente cortos, uno de ellos es simplemente un punto y los otros se componen por 3 datos además de provenir de la misma serie.

En la figura 9 encontramos la representación reducida de los datos. Encontramos los resultados de aplicar el método para la presión (rojo), el caudal (verde) y la combinación de ambas (azul). Para las variables por separado, observamos nubes de puntos dispersas sin ninguna estructura. Cuando combinamos las variables, la representación en la componente principal cae casi a 0. Es posible que el shapelet que determine la posición en la componente 1 no se encuentre en muchas de las series.

Por otro lado, el coste computacional de este algoritmo es especialmente alto. El último modelo tarda más de 45 minutos en generarse. Esto se debe al coste de buscar y definir cada shaplet tienendo un total de $d = 48$ dimensiones por registro.

Todas las evidencias apuntan a que la estructura de los datos no es visible aplicando shapelets. Comprobemos ahora, el método lineal por excelencia, PCA.

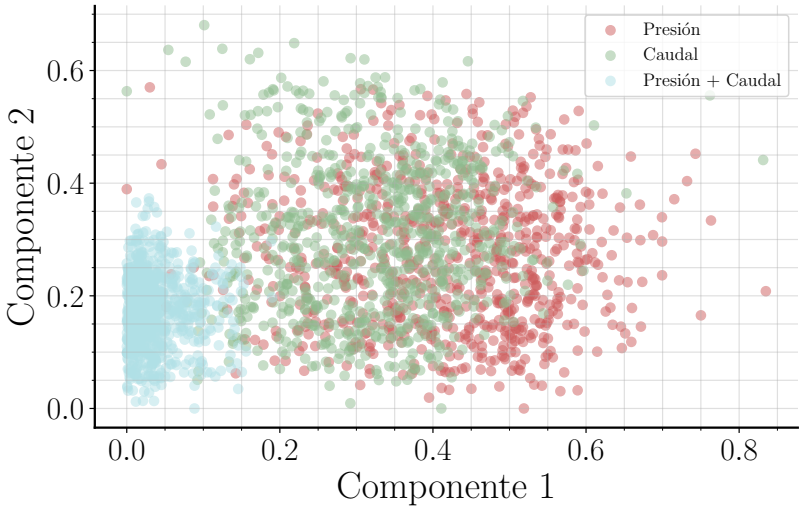


Figura 9: Representación de los datos por el método de shapelets. Cada color indica una variable distinta. Observamos que son nubes de puntos sin estructura.

7.1.2. PCA

Combinamos las dos variables, aplicamos PCA y extraemos las primeras 3 componentes principales. Contrastamos la representación con meses del año además de la semana.

En la figura 12a podemos ver que existe una división de los datos por la mitad y que cada mitad contiene los consumos de días laborables, sábados y domingos. Estos 3 grupos son distinguibles en la gráfica.

En la figura 12b encontramos la división más clara, observamos que está relacionado con la época del año. En los meses de verano (nube de la derecha), cuando las temperaturas son mayores el consumo de agua tanto para la higiene como personal en zonas de viviendas provoca perfiles de consumo distintos. A la izquierda, los meses entre septiembre y marzo.

Observamos la distribución de los datos en la componente 1, para los días de semana (10) y la componenete principal para los meses del año (11). En ambos encontramos que las distribuciones están desplazadas y sus formas difieren. Para el caso de los meses, observamos distribuciones en la zona de la izquierda para los meses entre abril y julio mientras, agosto, septiembre y octubre guardan distribuciones en dos zonas mientras que los meses de invierno tienen un valor máximo a la izquierda. Encontramos que PCA hace distinción en la época del año (primera CP) y día de la semana (segunda CP).

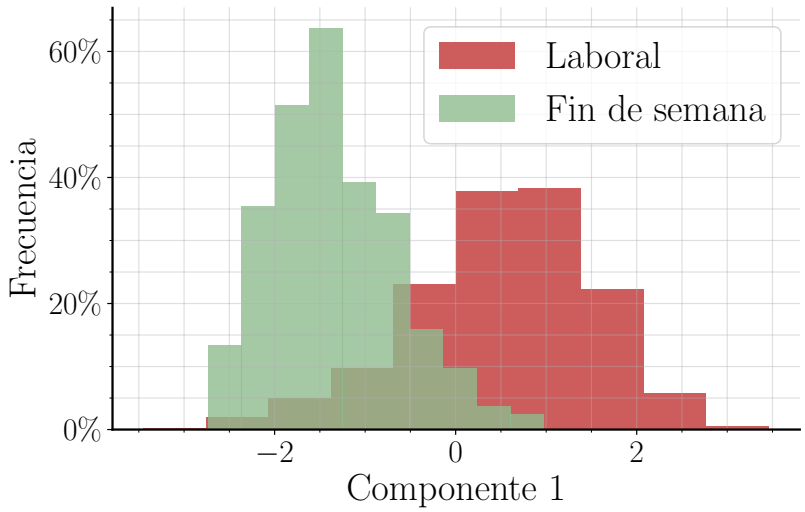


Figura 10: Distribución de los datos en la dirección de la componente principal por cada mes.

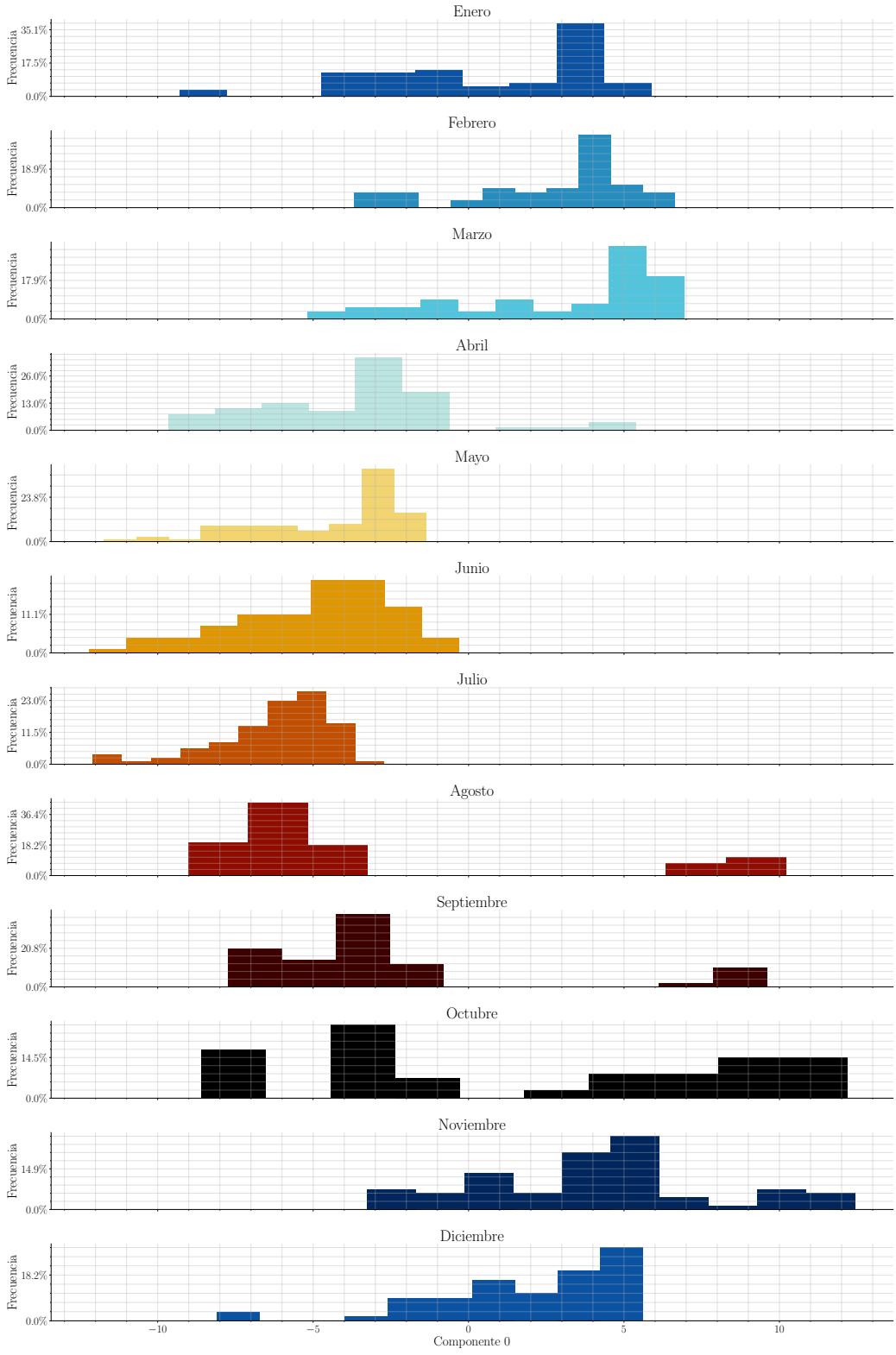
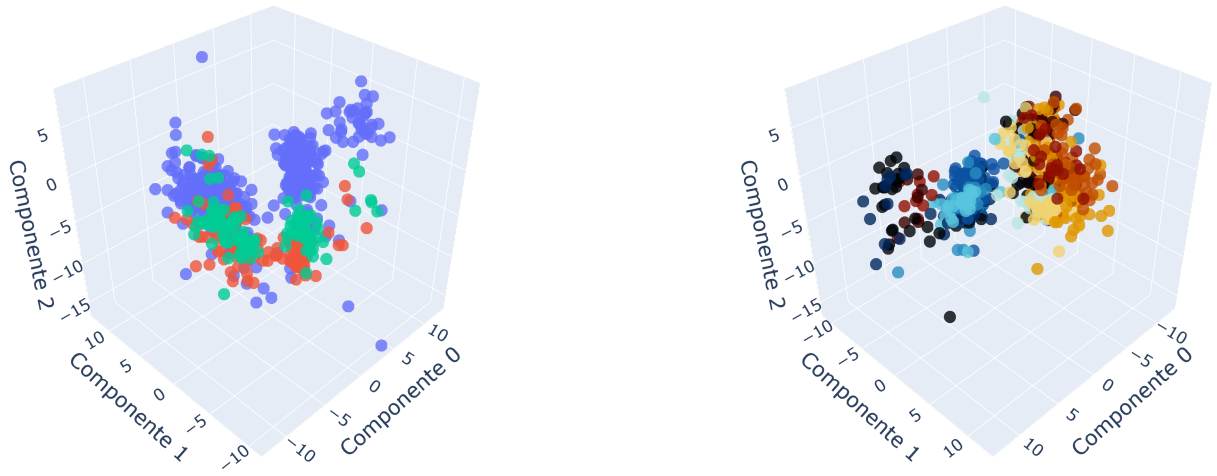


Figura 11: Distribución de los datos en la dirección de la componente principal por cada mes.



(a) Representación PCA de los datos en $d = 3$. Los colores representan distintos días de la semana. Azul son días laborales, verde son los sábados y rojo los domingos.

(b) Representación PCA de los datos en $d = 3$. Los colores representan distintos meses del año.

Figura 12

7.1.3. LLE

Elegimos 15 vecinos cercanos para generar un mapeado a $d = 3$ utilizando el modelo de LLE. En la figura 13 los datos se posicionan sobre caminos y aparece regiones de muy baja densidad, esta es la estructura típica que surge de aplicar LLE [20].

Una rama recoge datos de meses cálidos mientras que la otra contiene registro de los meses fríos. Los datos pertenecientes a octubre y septiembre parecen encontrarse en la región media. La información de cada rama es limitada, se encuentran proyectados en una línea. Otros métodos hacen mejor provecho del espacio para representar las estructuras.

7.1.4. Isomapas

Los isomapas producen una nube de puntos (fig14) extendida sobre la primera componente. Los meses cálidos se encuentran en la región negativa y los fríos, en la región positiva. Observando la figura, las otras componentes no parecen señalar ninguna estructura característica de los datos.

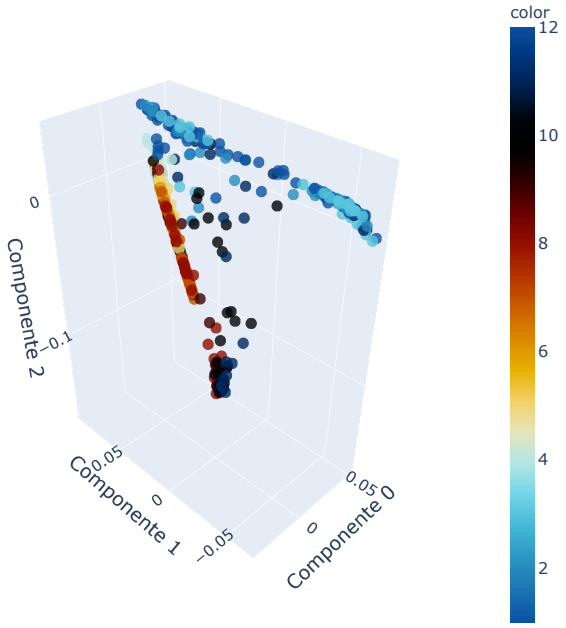


Figura 13: Resultado de aplicar LLE para la reducción de la dimensión para 15 primeros vecinos.

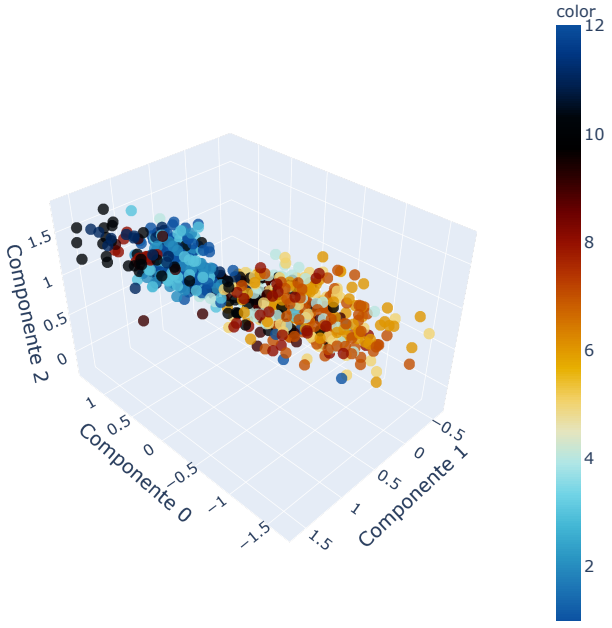


Figura 14: Método de isomapas aplicado para 15 vecinos. Los colores representan el mes que fue recogido el punto.

7.1.5. t-SNE

El algoritmo t-SNE genera la figura 15. Los hiperparámetros se determinan de forma visual. t-SNE es especialmente sensible a la perplejidad. Observamos una distinción clara entre el

consumo de verano y de invierno.

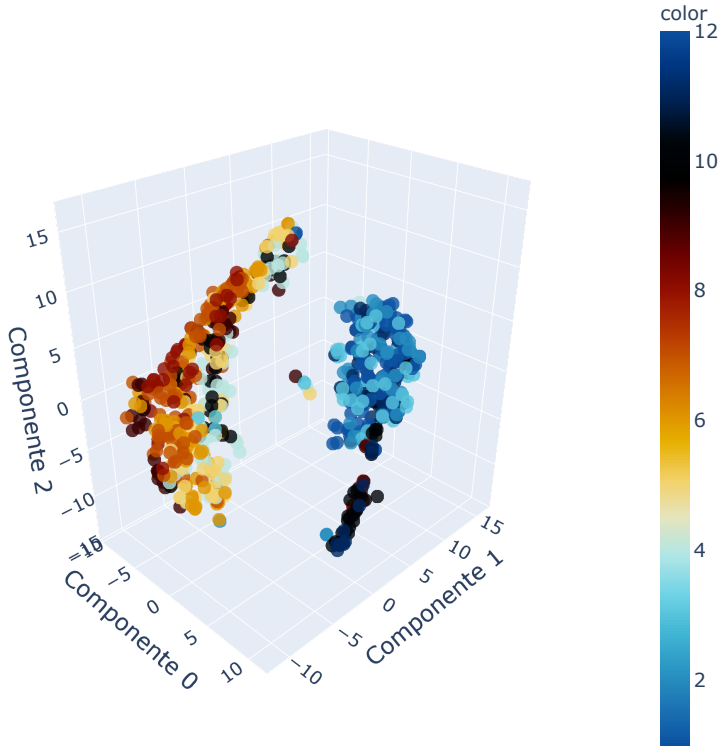
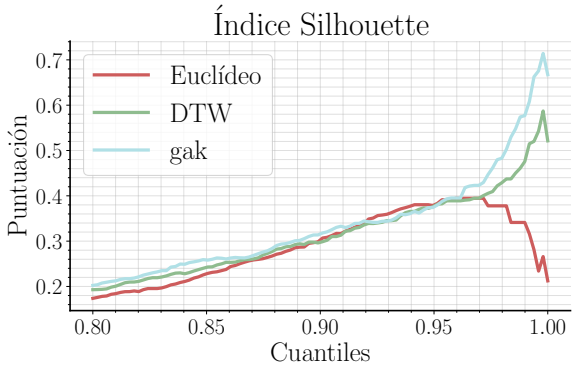


Figura 15: Representación de los datos utilizando t-SNE con ritmo de aprendizaje 100 y perplejidad 20.

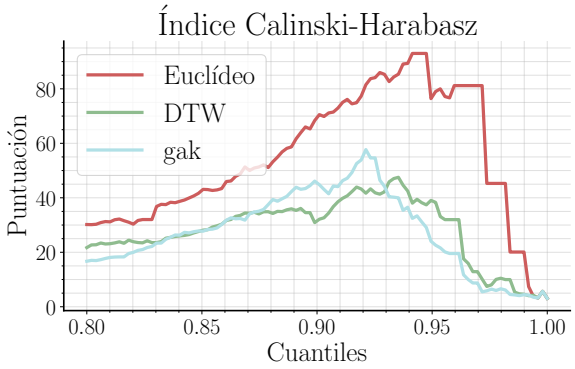
Además, la región de meses fríos se subdivide en otras 3 regiones, dos de ellas aparecen solapadas en la figura y la última se encuentra debajo. Visualmente se determina que este es el método que mejor separa los datos y los reagrupa según el grado de similitud.

7.2. Detección de anomalías

Para la detección de anomalías aplicamos los índices de Silhouette y de Calinski. Para OPTICS y LOF se varían los hiperparámetros del umbral de alcance (OPTICS), el número de vecinos y el grado de contaminación (LOF). Para OPTICS se obtienen las siguientes figuras (16).



(a) Índice Silhouette para varios umbrales de alcance según el cuartil de datos con un alcance por debajo de este umbral. Cada línea reresenta una métrica distinta.



(b) Índice Calinski para varios umbrales de alcance según el cuantil de datos con un alcance por debajo de este umbral. Cada línea representa una métrica distinta

Figura 16

Observamos que para ambos índice se produce un pico en las puntuaciones. Este máximo coincide en el caso de la métrica DTW y GAK para Silhouette y las métricas euclídea y GAK para Calinski. En la figura 16a, el pico para la métrica euclídea se encuentra mucho antes que las otras, lo que implica que en esta métrica se detectan más datos anómalos (alrededor de 4 % del total de registros). La forma de las curvas son muy distintas entre ambos índices, las curvas en 16a son más abruptas y señalan a un número pequeño de anomalías.

Para LOF obtenemos la figura 17. Observamos que para las 3 métricas con un grado de contaminación por debajo de 0,1, existe un máximo en los índices para `nn`= 50. Para número de vecinos muy bajo (alrededor de 5) observamos otro máximo en 17b y 17c.

En las tablas (1) aparecen las puntuaciones de los métodos optimizados con los datos. Para determinar la mejor métrica y método para aplicar a los datos, comparamos los valores de ambas puntuaciones.

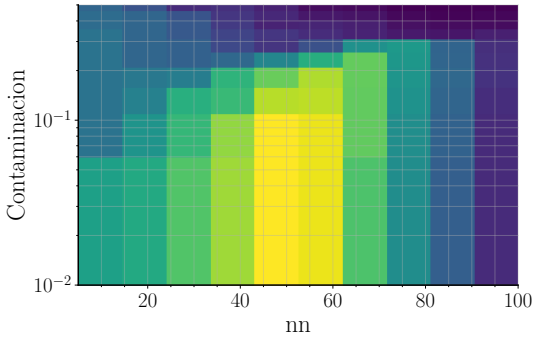
	BDSCAN	OPTICS	LOF
Euclídeo	0.359	0.380	0.358
DTW	0.383	0.587	0.336
GAK	0.313	0.714	0.347

(a) Resultados de la puntuación de Silhouette para los distintos algoritmos de detección de anomalías.

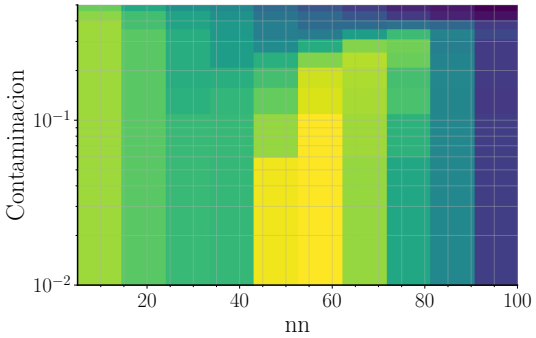
	BDSCAN	OPTICS	LOF
Euclídeo	79.7	93.0	57.5
DTW	22.4	5.67	33.5
GAK	47.0	5.67	34.9

(b) Resultados de la puntuación de Calinski para los distintos algoritmos de detección de anomalías.

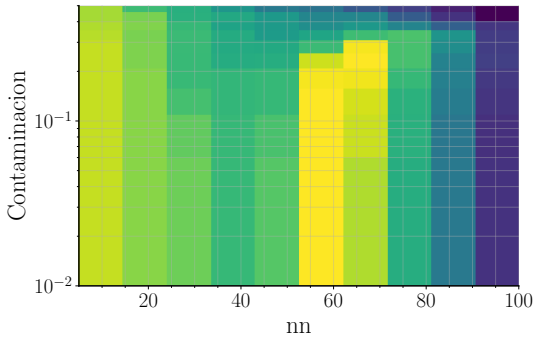
Tabla 1



(a) Espacio de los hiperparámetros `contamination` y `nn` para la métrica euclídea. Cuanto más claro es el color, mayor es la puntuación para LOF con esos valores.



(b) Espacio de los hiperparámetros `contamination` y `nn` para la métrica DTW. Cuanto más claro es el color, mayor es la puntuación para LOF con esos valores.



(c) Espacio de los hiperparámetros `contamination` y `nn` para la métrica euclídea. Cuanto más claro es el color, mayor es la puntuación para LOF con esos valores.

Figura 17

La métrica euclídea es la más robusta de las 3, en 1b ofrece los mejores resultados para cualquier algoritmo mientras que en 1a los resultados son comparables a las otra métricas. OPTICS parece ser la mejor opción con esta métrica al tener la mejor puntuación en ambas tablas.

7.3. Métodos de clustering

A continuación presentamos los resultados de los algoritmos de HMM, K-means y BIRCH. En la tabla 2 aparece el resultado de escoger el mejor resultado de cada modelo.

Para HMM, creamos 6 clústers y producimos una visualización a partir de t-SNE para observarlos en $d = 3$ (fig 18). También representamos los estados ocultos como los puntos medios, al ser gaussianas, μ. Encontramos que el resultado diferencia entre grupos, además de incluir cada estado representativo del clúster (i.e. cada estado oculto) como un centroide en el espacio reducido.

Los resultados de K-means y BIRCH los encontramos en las figuras 19 y 20 respectivamente.

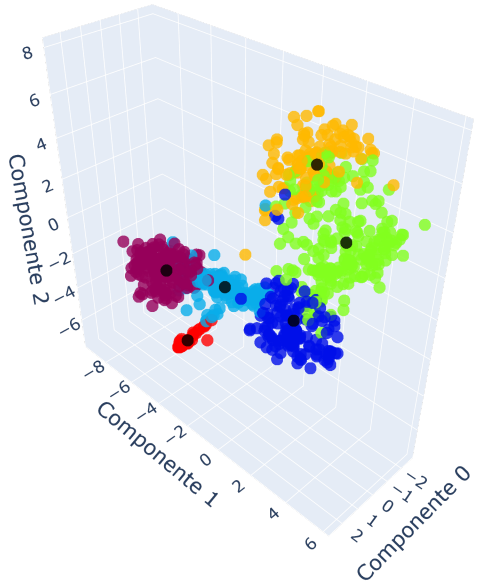


Figura 18: Representación reducida del modelo de Markov para $n = 6$ clústers. Los puntos negros representan los valores medios de cada estado oculto.

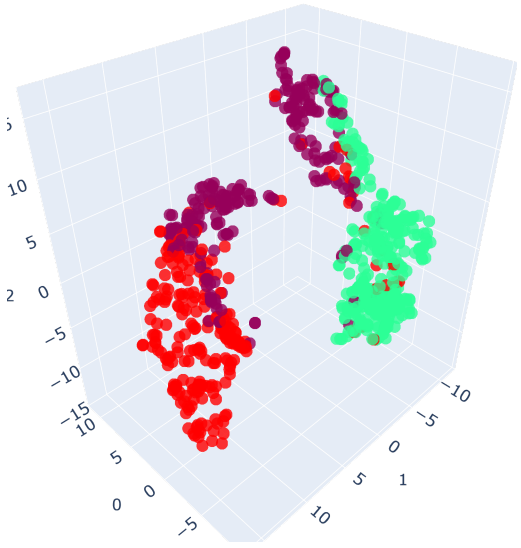
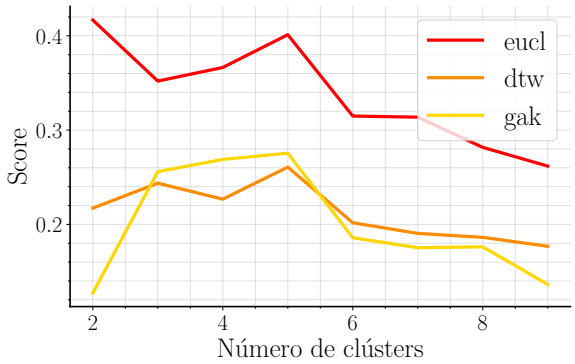
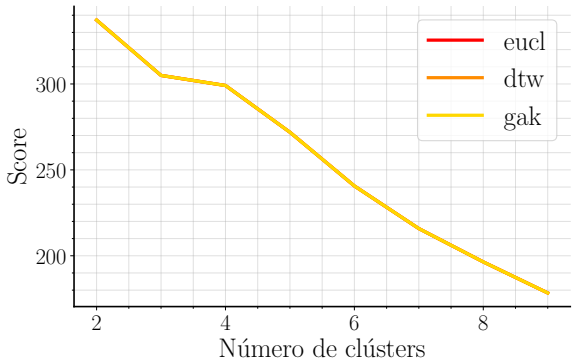


Figura 19: Representación reducida de K-means para $n = 3$ clústers, el número de agrupamientos óptimos para todas las métricas.

En K-means (19) hay una dispersión de una de las clases (roja) que provoca a algunos datos formar parte de las otras clases (derecha) en la representación reducida. Además, la clase púrpura se encuentra seccionada y adherida a las otras clases.



(a) Puntuación de Silhouette. Aparece un pico para $n = 5$ clases en todas las métricas.



(b) Puntuación de Calinski. Obsevamos un solapamiento de las tres métricas debido a que este método sólo acepta la distancia euclídea y de la misma forma, para la puntuación de Calinski.

Figura 20: Representación de las puntuaciones en el método BIRCH variando el número de clústers generados.

Los resultados del método Birch revelan un máximo para agrupaciones de 5 clases (20a). Recordemos que para K-means se hayaron 3 agrupaciones óptimas.

	HMM	K-means	BIRCH
Euclídeo	0.289	0.280	0.417
DTW	0.184	0.230	0.261
GAK	0.154	NaN	0.269

(a) Resultados de la puntuación de Silhouette para los distintos algoritmos de clustering.

	HMM	K-means	BIRCH
Euclídeo	235	590	337
DTW	235	362	272
GAK	235	NaN	299

(b) Resultados de la puntuación de Calinski para los distintos algoritmos de clustering.

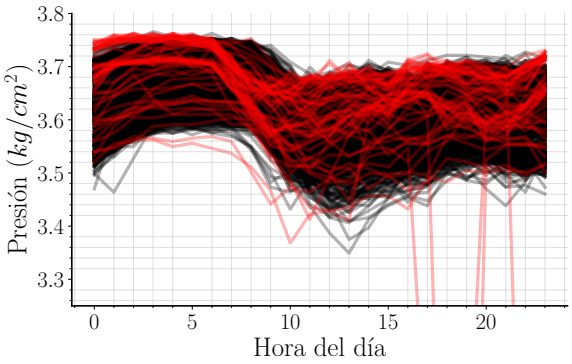
Tabla 2

Los resultados son consistentes con los obtenidos para los algoritmos de detección de anomalías (1) donde la métrica con mayor puntuación es la euclídea para todos los algoritmos y índices. Optamos por BIRCH como el método que mejor clasifica a los datos.

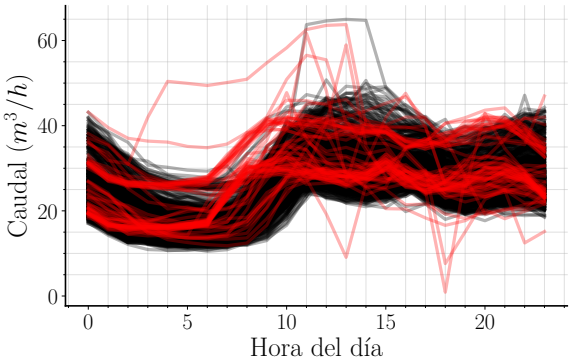
7.4. Método experimental

Creamos un modelo que incluya el algoritmo de OPTICS para detectar las anomalías, BIRCH para la clasificación del resto de datos y t-SNE para la visualización de los resultados. En este

caso, se emplearán todos los datos originales, en su estado previo a la eliminación de outliers visuales.



(a) Series temporales de la variable presión. Las series en rojo son detectadas como anomalías por el método OPTICS.



(b) Series temporales de la variable caudal. Las series en rojo son detectadas como anomalías por el método OPTICS.

Figura 21: Detección de anomalías por OPTICS en las series temporales de la presión y caudal.

En la figura 22 podemos observar la clasificación de los datos mediante BIRCH. Sin embargo, las anomalías parecen estar también agrupadas al contrario de nuestro concepto de anomalía donde esperamos encontrarlos mejor esparcidos. Además, podemos observar en 21 que una gran cantidad de las anomalías están recogidas en dos grupos visibles al inicio de cada serie (cerca de la hora 5).

Es posible que OPTICS detecte que este grupo sea pequeño y la distancia de alcance mayor para estos datos. Sin embargo, recordamos que LOF detecta estas variaciones en la densidad y se fija en anomalías dentro de los grupos. Comparamos los resultados aplicando LOF y obtenemos las siguientes figuras.

Podemos observar que las agrupaciones al inicio de las series (fig. 23a y 23b) se ha esparcido en todo el rango aunque todavía existen regiones donde se encuentran muchas anomalías. Podemos comprobarlo en la zona superior de la figura 23a.

La representación de las clases es muy similar al caso anterior. Sin embargo, se pueden observar algunas anomalías que antes no estaban siendo detectadas, la región azul en su parte superior en 24.

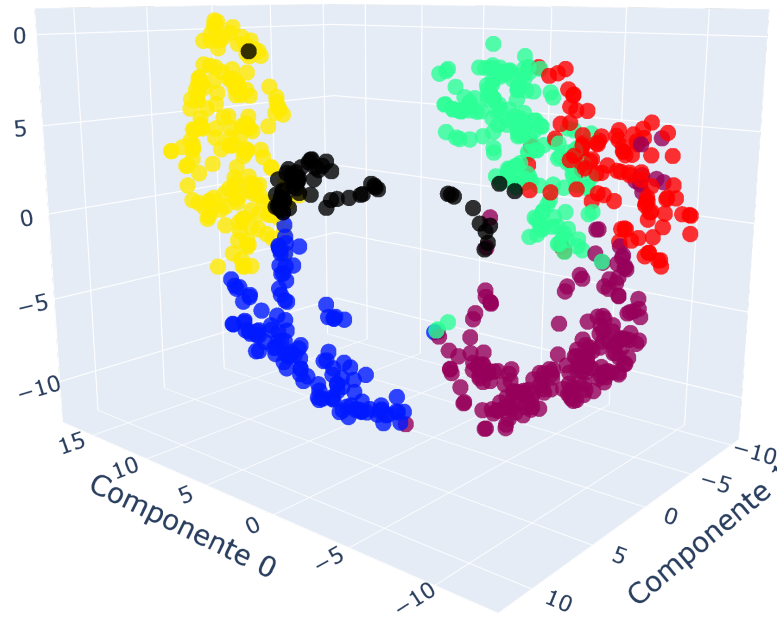
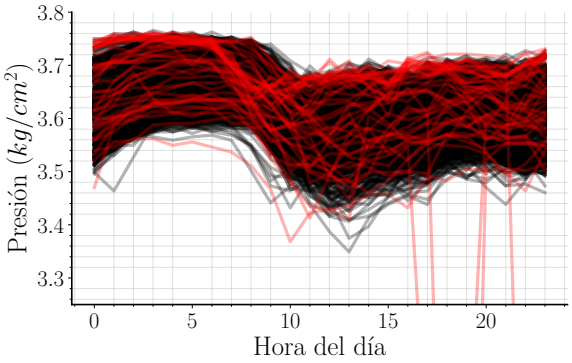
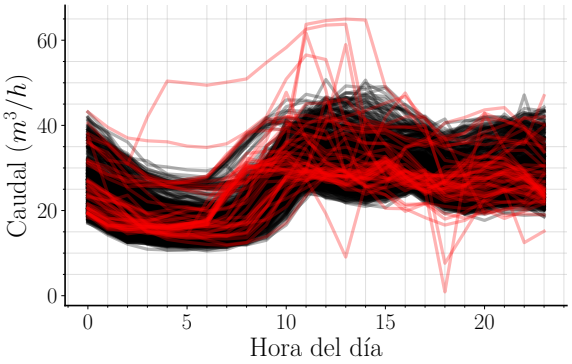


Figura 22: Representación reducida del método de clasficación de los datos. Los datos en negro representan los puntos anómalos detectados por OPTICS.



(a) Series temporales de la variable presión. Las series en rojo son detectadas como anomalías por el método LOF.



(b) Series temporales de la variable caudal. Las series en rojo son detectadas como anomalías por el método LOF.

Figura 23: Detección de anomalías por LOF en las series temporales de la presión y caudal

Aunque los índices han sido un apoyo para comparar y optimizar los métodos, un test visual es crítico en el análisis de series temporales no supervisadas.

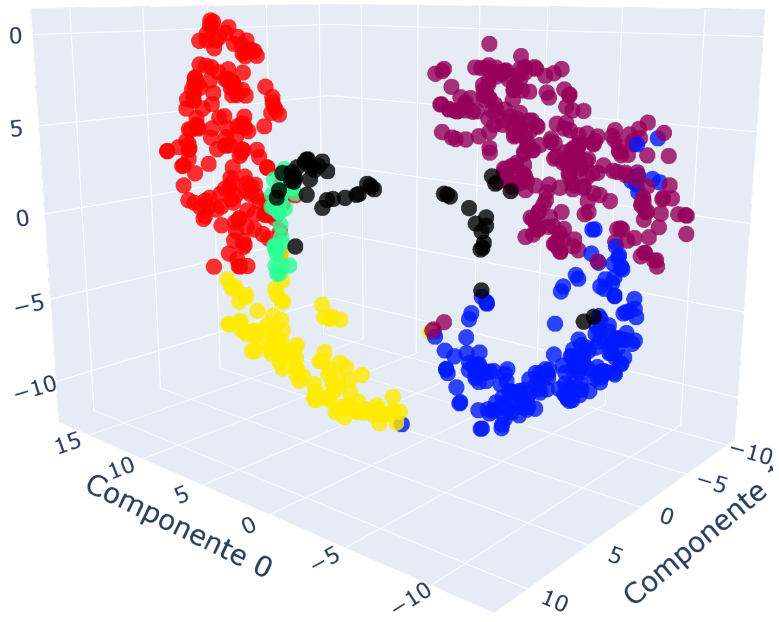


Figura 24: Representación reducida del método de clasficación de los datos. Los datos en negro representa los puntos anómalos detectados por LOF.

Figura 25: Representación reducida por LOF en las series temporales de la presión y caudal.

8. Conclusiones

En este proyecto hemos presentado algunos de los métodos de análisis de series temporales no supervisadas. Hemos estudiado los datos bajo visualizaciones de dimensionalidad reducida por varios algoritmos y observar qué efectos produce cada uno de ellos sobre los datos. El método t-SNE fue determinado como el algoritmo que mejor conserva la estructura de los datos. Además se emplearon métodos de detección de anomalías, con el objetivo de determinar el más efectivo para eliminar *outliers*, encontramos que OPTICS genera los mejores resultados. Aparte, se han clasificado los datos por distintos criterios de clustering y similitud hayando que BIRCH genera las mejores clasificaciones.

Se ha empleado el uso de índices para cuantificar la calidad de la detección de las anomalías y

clasificación de datos empleando el índice de Silhouette y de Calinski. También se ha observado los resultados cambiando a distintas métricas obteniendo la euclídea como la más robusta.

Finalmente se ha combinado los algoritmos que mejor realizaron su tarea para obtener una mejor imagen de los datos. En este proceso se demostró que el uso de LOF es más efectivo que OPTICS para la detección de las anomalías debido a la capacidad de detectar agrupaciones de baja densidad. Esto además demostró los límites que presenta el uso de los índices para comparar métodos.

Referencias

[1] S. Aghabozorgi, A. S. Shirkhorshidi, and T. Y. Wah. Time-series clustering – a decade review. *Information Systems*, 53:16–38, oct 2015. doi: 10.1016/j.is.2015.04.007. URL https://doi.org/10.1016%2Fj.is.2015.04.007.

[2] J. Alonso del Saso. Dataset of normalized 24-hour vectors. https://doi.org/10.5281/zenodo.4022906, 2020.

[3] J. Alonso del Saso. TFM2020. https://github.com/javialonsaso/TFM2020, 2020.

[4] J. A. Bilmes et al. A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. 1998. URL http://www.super.tka4.org/materials/lib/Articles-Books/Speech%20Recognition/A%20Gentle%20Tutorial%20of%20the%20EM%20Algorithm.pdf.

[5] T. Caliński and J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3(1):1–27, 1974. doi: 10.1080/03610927408827101. URL https://www.tandfonline.com/doi/abs/10.1080/03610927408827101.

[6] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), July 2009. ISSN 0360-0300. doi: 10.1145/1541880.1541882. URL https://doi.org/10.1145/1541880.1541882.

[7] A. Claudia and L. O. Arlindo. Temporal data mining: an overview. In *Proceedings of KDD Workshop on Temporal Data Mining*, pages 1–13, 2001. URL https://pdfs.semanticscholar.org/f7b5/9a929cc1304ef2fbde4a08992c25b34404d3.pdf.

[8] M. Cuturi. Fast global alignment kernels. In L. Getoor and T. Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 929–936, New York, NY, USA, June 2011. ACM. ISBN 978-1-4503-0619-5. URL https://icml.cc/Conferences/2011/papers.php.html.

[9] M. Ester, H. peter Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *In proceedings of the 2nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 226–231. AAAI Press, 1996. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.407.5381.

[10] J. Grabocka, N. Schilling, M. Wistuba, and L. Schmidt-Thieme. Learning time-series shapelets. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, page 392–401, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450329569. doi: 10.1145/2623330.2623613. URL https://doi.org/10.1145/2623330.2623613.

[11] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95. doi: 10.1109/MCSE.2007.55. URL https://doi.org/10.5281/zenodo.3984190.

[12] I. T. Jolliffe. *Principal Components in Regression Analysis*, pages 129–155. Springer New York, New York, NY, 1986. ISBN 978-1-4757-1904-8. doi: 10.1007/978-1-4757-1904-8_8. URL https://doi.org/10.1007/978-1-4757-1904-8_8.

[13] E. Keogh and J. Lin. Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowledge and Information Systems*, 8(2):154–177, aug 2005. doi: 10.1007/s10115-004-0172-7. URL https://doi.org/10.1007%2Fs10115-004-0172-7.

[14] E. J. Keogh. Exact indexing of dynamic time warping. In *Proceedings of 28th International Conference on Very Large Data Bases, VLDB 2002, Hong Kong, August 20-23, 2002*, pages 406–417. Morgan Kaufmann, 2002. doi: 10.1016/B978-155860869-6/50043-3. URL http://www.vldb.org/conf/2002/S12P01.pdf.

[15] T. E. Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[16] T. pandas development team. pandas-dev/pandas: Pandas, Feb. 2020. URL https://doi.org/10.5281/zenodo.3509134.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[18] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria,

and E. Keogh. Addressing big data time series: Mining trillions of time series subsequences under dynamic time warping. *ACM Trans. Knowl. Discov. Data*, 7(3), Sept. 2013. ISSN 1556-4681. doi: 10.1145/2500489. URL https://doi.org/10.1145/2500489.

[19] P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, nov 1987. doi: 10.1016/0377-0427(87)90125-7. URL https://doi.org/10.1016%2F0377-0427%2887%2990125-7.

[20] S. T. Roweis. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, dec 2000. doi: 10.1126/science.290.5500.2323. URL https://doi.org/10.1126%2Fscience.290.5500.2323.

[21] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu. DBSCAN revisited, revisited. *ACM Transactions on Database Systems*, 42(3):1–21, Aug 2017. doi: 10.1145/3068335. URL https://doi.org/10.1145%2F3068335.

[22] R. Tavenard, J. Faouzi, G. Vandewiele, F. Divo, G. Androz, C. Holtz, M. Payne, R. Yurchak, M. Rußwurm, K. Kolar, and E. Woods. Tslearn, a machine learning toolkit for time series data. *Journal of Machine Learning Research*, 21(118):1–6, 2020. URL http://jmlr.org/papers/v21/20-091.html.

[23] J. B. Tenenbaum. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, dec 2000. doi: 10.1126/science.290.5500.2319. URL https://doi.org/10.1126%2Fscience.290.5500.2319.

[24] L. van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–26050, nov 2008. URL http://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf.

[25] S. Van Der Walt, S. C. Colbert, and G. Varoquaux. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22, 2011.

[26] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH. In *Proceedings of the 1996 ACM SIGMOD international conference on Management of data - SIGMOD '96*. ACM Press, 1996. doi: 10.1145/233269.233324. URL https://doi.org/10.1145%2F233269.233324.