Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Análisis y comparación de algoritmos de detección de anomalías

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Jorge Zaragoza Gauchía

Tutor: Jon Ander Gómez Adrian

Curso 2019-2020

Resumen

El principal objetivo de este estudio es analizar el comportamiento de varios algoritmos de detección de anomalías. Se estudiarán distintas librerías del lenguaje Python en la que se encuentran los algoritmos, que se aplicarán a distintos conjuntos de datos de diferentes campos de la ciencia.

Se trata de analizar y comparar la eficacia de dichos algoritmos para cada tipo de conjuntos de datos. Teniendo en cuenta la diversidad de campos de aplicación de la detección de anomalías el objetivo del trabajo es el análisis de los algoritmos mediante distintas métricas independientemente del tipo de conjunto de datos. Se compararán generalmente las técnicas actuales de detección de comportamientos anómalos y se examinarán las distintas técnicas de análisis y captura de anomalías.

Palabras clave: Análisis de datos, Detección de anomalías, Python, Aprendizaje automático, Conjunto de datos.

Abstract

The main objective of this paper is to analyze the behavior of different algorithms for anomaly detection. We will study different libraries of Python language in which different algorithms are found, which will be applied to different data sets from different fields of science.

The aim is to analyze and compare the effectiveness of these algorithms for each type of data set. Considering the diversity of fields of application of anomaly detection, the objective of the work is the analysis of the algorithms using different metrics regardless of the type of data set. Current techniques for detecting abnormal behavior will generally be compared and the different techniques for analyzing and capturing anomalies will be examined.

Keywords: Data Analysis, Anomaly detection, Python, Machine Learning, Dataset

Tabla de contenidos

CAPÍTULO 1: Introducción

En el mundo actual conceptos como inteligencia artificial, análisis de datos y aprendizaje automático se encuentran presentes en el día a día en cualquier tipo de negocio. La detección de anomalías es un campo de la inteligencia artificial que usa métodos de aprendizaje automático para la detección de elementos o sucesos qué no se ajustan a una pauta esperada.

1.1 Motivación

La selección del trabajo está motivada en parte por la elección de la rama sistemas de la información en el plan de estudios del grado, el cursado de ciertas asignaturas más aproximadas a la computación como la minería de datos durante los estudios de Erasmus, y una primera experiencia laboral en el campo de SAP en el mundo de la consultoría. Haber cursado esas asignaturas y mi primera experiencia laboral despertaron en mí un gran interés en el campo del aprendizaje automático y la detección de anomalías, ya que el mismo está estrechamente ligado a mis inquietudes con la estadística, así como una aproximación a las asignaturas cursadas en los estudios del Erasmus.

Con la mirada puesta en un futuro laboral en el campo de la inteligencia artificial y el análisis de datos, que crece día tras día, y con el objetivo de ser una aportación más a una comunidad que considero de interés para la sociedad, se decide abordar un tema de gran importancia como es el uso de los algoritmos en distintos conjuntos de datos y su evaluación.

Mediante los consejos de grandes profesionales en este ámbito se decide la investigación sobre estos algoritmos de detección de anomalías en el lenguaje Python. Tanto el funcionamiento como los resultados de los algoritmos se consideran de gran interés debido a la variedad de campos en los que son aplicables.

En una sociedad en la que la informática se especializa día tras día, la diversidad de aplicaciones que proporciona la detección de anomalías en una multitud de campos genera un gran interés en mi persona y en mi opinión es un campo de interés común.

1.2 Objetivos

El objetivo de este trabajo de fin de grado es el análisis de los algoritmos de aprendizaje automático no supervisados para la detección de anomalías.

Se busca llegar a una conclusión sobre la eficacia de la aplicación de los algoritmos en los distintos conjuntos de datos a través de la obtención de resultados comparados por distintas métricas y tipos de algoritmos. Se realizará tanto el análisis por separado de cada conjunto de datos como el análisis global con el propósito de mejorar el estado del arte actual.

Para ello partiendo de técnicas de aprendizaje no supervisado y conjuntos de datos de distintos campos se busca llegar a una relación entre estos y su funcionamiento mediante la evaluación del rendimiento a través de diversas métricas.

La selección de algoritmos se basa en la más documentada de la bibliografía valorando distintos tipos como son basados en vecinos, basados en agrupación, probabilísticos o subespaciales.

Mediante librerías y paquetes externos como Pyod o Scikit Learn se usarán las métricas para análisis de resultados más frecuentados en la bibliografía con el objetivo de sintetizar este análisis y proporcionar respuestas claras al uso de qué tipo de algoritmo en cada caso.

1.3 Metodología

Investigación en el campo de la inteligencia artificial mediante numerosos artículos de towardsdatascience, diversos estudios online y publicaciones en portales: Analythics Vidhya, KDnuggets. Y lectura para aproximación teórica al funcionamiento de los algoritmos y comprensión del estado del arte mediante el libro "Terrorism, Security and Computation" de VS.Subrahmanian y "Introduction to Machine Learning" de Ethem Alpaydin .

Análisis y selección de conjuntos de datos mediante lectura de conjuntos de datos (*datasets*) más usados y referenciados en el campo del aprendizaje automático a través de la librería ODDS. Selección y comprensión de los más útiles para la ejecución de nuestro análisis.

Ejecución de código mediante lenguaje Python con el uso principal de las librerías, Pyod y Scikit Learn.

En cuanto a las dificultades con el código, el foro stackoverflow ha resuelto dudas y clarificado conceptos.

1.4 Estructura de la memoria

La memoria se divide en 7 capítulos que se resumen a continuación:

En el Capítulo 1 de Introducción se habla de la motivación y objetivos del trabajo, así como de la metodología que indica las bases de información para el trabajo.

En el Capítulo 2 se menciona el estado del arte, así como una primera introducción al concepto de anomalía, ejemplos de uso y aplicaciones prácticas en la actualidad y futuro.

En el Capítulo 3 se trata el conjunto de datos, indicando una aproximación teórica de que es, el porqué de su selección y el proceso que debe seguir para la ejecución del código en este trabajo.

En el Capítulo 4 se comentan los diversos tipos de aprendizaje automático que existen, profundizando el no supervisado y sus algoritmos. Se clasifican y se explican teóricamente en cuanto a funcionamiento.

En el Capítulo 5 de ejecución se muestran las fases de ejecución del código y se muestran los resultados obtenidos para la continuación de las fases.

En el Capítulo 6 se realiza un análisis de los resultados obtenidos para cada conjunto de datos y de forma global.

En el Capítulo 7 se encuentra la conclusión.

CAPÍTULO 2: Detección de anomalías

2.1 Estado del arte

La detección de anomalías a través de la inteligencia artificial (IA) ayudará a una empresa a identificar problemas las 24 horas del día de forma más rápida y fiable, aumentando la eficiencia y el rendimiento [1].

Gran cantidad de artículos recalcan la necesidad de la detección de anomalías en los negocios. Dejando para futuros apartados del trabajo los distintos campos y aplicaciones en este apartado se observa el interés que tiene la aplicación de detección de anomalías desde la pequeña a la gran empresa.

Se dice que la información es poder, y cada vez se tiene más en cuenta que esa información en la sociedad actual viene dada por los datos. Ahora bien, una gran cantidad de datos conlleva poder si se manejan correctamente.

La detección de anomalías es clave para las empresas, ya que su supervivencia puede venir determinada por la misma. La detección en una fase temprana es un factor importante. En un gran número de ocasiones la detección tarda en detectarse semanas, meses o incluso años, y al revisar las empresas los balances e indicadores de negocio, estas anomalías podrían haber causado ya estragos.

Según un artículo de Forbes [2] el 61% de los vendedores planean usar el aprendizaje automático como parte de su estrategia de datos, dado que todavía hay empresas que se están perdiendo esta ventaja con el resto de los competidores. Se remarca el hecho de que, entre otros, pueden ayudar a descubrir palabras clave y otros elementos de las campañas de marketing que no se están aprovechando, prevenir las violaciones y amenazas a la seguridad y detectar amenazas y problemas antes de que causen daños.

En el mismo estudio de Forbes se menciona el caso de la empresa de consultoría Accenture. Casi el 10% de sus 25 millones de procesos anuales de líneas de gastos estaban siendo marcados por incumplimiento o fraude. Mientras que su sistema basado en reglas funcionaba hasta cierto punto, Accenture implementó un algoritmo de aprendizaje automático para optimizar el proceso. Se utilizó para reducir los falsos positivos, detectar los valores anómalos y crear una solución no supervisada.

Por supuesto es difícil saber que pasa exactamente con los datos, pero es ahí donde entra la inteligencia artificial. Viendo que el 85% de las empresas están de acuerdo en que la IA les ayudará a obtener una ventaja competitiva, la IA se está convirtiendo en una tecnología que las empresas adoptan cada vez más [3]. Y este creciente interés va acompañado de la adopción cada vez mayor de herramientas de detección de anomalías.

Herramientas como por ejemplo Google Analytics, Facebook Ads y Shopify no son capaces de abordar todos los datos en grandes empresas. Y es aquí donde un negocio debe apostar por mecanismos de detección de anomalías con algoritmos de aprendizaje automático.

Durante el trabajo se revisan constantemente artículos relacionados con avances y usos de la detección de anomalías, es ahí donde se puede ver el verdadero alcance de este método. Desde herramientas para videojuegos, detección de blanqueo de Bitcoin, conducción automática de

vehículos, detección de anomalías en imágenes de rayos X y un gran número más de distintas aplicaciones [4].

2.2 Anomalía: Definición.

Como primera aproximación al termino se cita la definición de la RAE [5] en la cual se destacan las dos primeras acepciones.

1. f. Desviación o discrepancia de una regla o de un uso.
2. f. Defecto de forma o de funcionamiento.

La detección de anomalías es la técnica para identificar eventos raros u observaciones que se pueden generar al ser estadísticamente diferentes del resto de las observaciones en los datos. Tal comportamiento anómalo generalmente se traduce en algún tipo de problema.

La detección de anomalías cuenta con distintos enfoques que se basan en modelos y predicciones de datos pasados. Una suposición principal del comportamiento normal es la estacionalidad de los datos, es decir, que se supone que los procesos que llevaron a cabo la generación de datos no han cambiado. Por lo tanto, lo que se ha observado en los datos es lo que se espera ver en el futuro. La naturaleza de los atributos determina la aplicabilidad de las técnicas de detección de anomalías.

Una anomalía dentro del campo de la Inteligencia Artificial puede clasificarse ampliamente en tres categorías [6]:

- Anomalía puntual:

Corresponden a anomalías de datos individuales que pueden considerarse anómalos con respecto al resto de datos. Cómo se muestra en la Figura 2.1

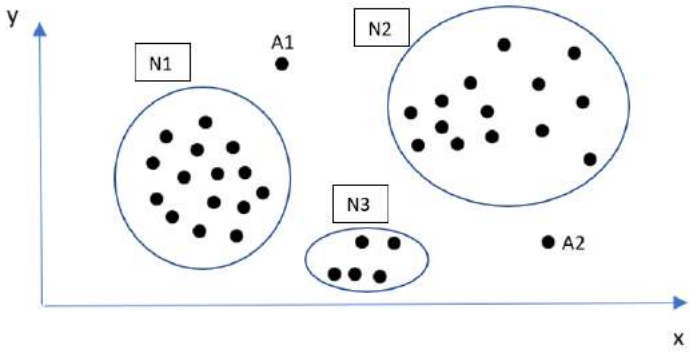

Figura 2.1 Grafico bidimensional de anomalías puntuales [7].

- Anomalía contextual:

Es una anomalía contextual si es una anomalía debido al contexto de la observación. Este contexto es inducido por cierta estructura del conjunto de datos y debe ser una especificación del problema.

Para cada instancia de los datos se debe definir tanto un atributo contextual como un atributo de comportamiento. Se exploran con mayor frecuencia en conjuntos de datos de series temporales.

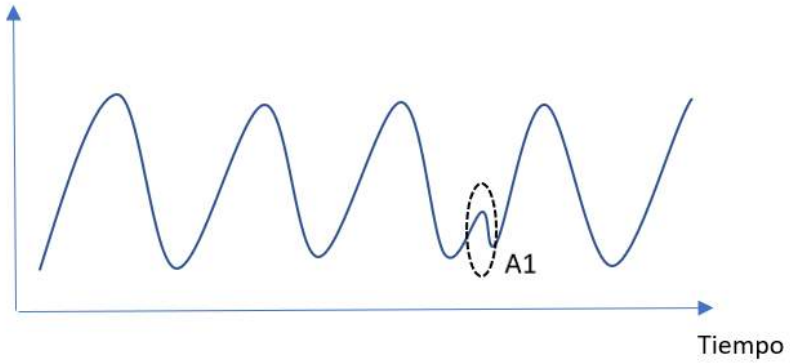

Figura 2.2 Grafico anomalía contextual con el paso del tiempo [7].

- Anomalía colectiva:

Es una anomalía colectiva si una colección de instancias es anómala con respecto a todo el conjunto de datos. Un conjunto de instancias de datos ayuda a encontrar una anomalía. Existe la posibilidad de que datos individuales no sean una anomalía, pero si en su conjunto. Se muestra en la figura 2.3 obtenida de [8].

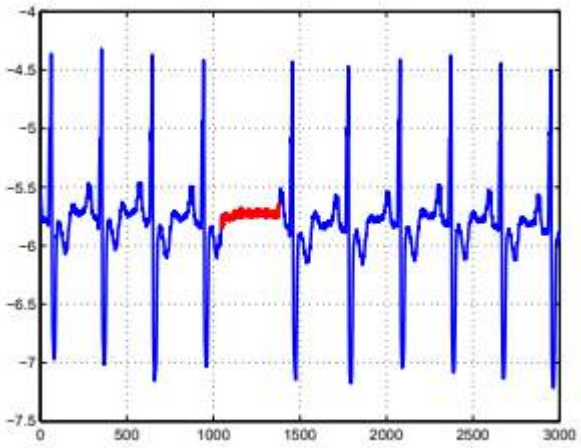

Figura 2.3 Salida de electrocardiograma humano con anomalía colectiva.

Es necesario comprender que estas anomalías pueden estar conectadas y no forman parte de un único tipo de anomalía. La anomalía puntual es capaz de volverse contextual si se le aplica cierto contexto. De igual forma las anomalías puntuales podrían convertirse en colectivas, si unimos múltiples anomalías puntuales.

Entre los desafíos que representan las tareas de detección de anomalías se plantean situaciones como que con el paso del tiempo el comportamiento cambia, y lo que se considera normal hoy en día no tiene porqué serlo en el futuro. Los enfoques de aproximación son muy distintos en los diversos campos y en un gran número de ocasiones resulta de gran dificultad el hecho de considerar un dato como normal o anomalía.

2.3 Aplicaciones prácticas

En los siguientes puntos se mencionan casos en distintas áreas de aplicación en las que la detección de anomalías es útil y a veces necesaria para realizar ciertas tareas criticas [9].

2.3.1 Salud

La salud es un dominio en el que la inteligencia artificial avanza a grandes pasos, muchos aspectos de la salud humana pueden verse analizados por la detección de anomalías. Dos de las áreas principales son las de diagnóstico y monitorización de pacientes. A continuación, se tratan en mayor profundidad algunos ejemplos:

- Monitorización de pacientes.
- Radiología: Detección de planes de tratamiento erróneos en los datos de radioterapia de series temporales
- Epidemiología: La ciencia médica también puede beneficiarse identificando los puntos en el tiempo en que los datos epidemiológicos revelan que los medicamentos que anteriormente habían tenido éxito dejan de ser útiles para los pacientes, lo que significa la aparición de una mutación resistente a los medicamentos del patógeno responsable. También pueden obtenerse datos de pacientes individuales, cuya respuesta a un medicamento puede seguir un camino inusual, por ejemplo, que parece mejorar y luego se degrada rápidamente.

2.3.2 Industria

En la industria este tipo de algoritmos pueden ser de gran ayuda ya que pueden detectar los fallos en fases tempranas de los procesos industriales.

- Ventas al por menor
- Gestión de inventario
- Comportamiento del cliente y de los empleados.
- Control de calidad: puede supervisar y ayudar a realizar el mantenimiento de la maquinaria de producción, reprogramar los ordenadores industriales (inteligencia distribuida) para la producción de nuevos productos y optimizar la eficiencia de las operaciones de la planta en toda la cadena de suministro.

2.3.3 Finanzas

En el campo de las finanzas existen diversas aplicaciones a la detección de anomalías. Muchos ataques a través de la red o crímenes pueden ser evitados gracias a estos modelos. Algunos ejemplos de esto son:

- Fraude de tarjeta de crédito
- Detección de solvencia para créditos
- Predicción de bancarrota
- Inversión en bolsa: La aplicación de algoritmos de detección de anomalías puede proporcionar información valiosa a los inversores potenciales y actuales de la empresa. La predicción exacta es prácticamente imposible, sin embargo, el rendimiento de una acción durante un período corto (reciente) puede compararse con su rendimiento anterior, así como con el rendimiento de otras acciones de empresas similares, lo que puede ayudar a identificar anomalías que puedan significar que la empresa tiene un rendimiento superior o inferior al de sus competidores.

CAPÍTULO 3: Conjunto de datos

3.1 Definición

Conjunto de datos: Una colección de conjuntos de información separados que son tratados como una sola unidad por una computadora [10].

El término **dataset** es una representación de datos conocido en español como conjunto de datos o serie de datos. Esta representación viene dada en una única tabla de base de datos o matriz de datos y se podría definir como una representación de datos residentes en memoria.

El conjunto de datos se almacena por columnas y filas, siendo cada columna una variable (atributo como color, talla...) y cada fila representa a un miembro determinado del conjunto de datos (en el caso de color amarillo, en el caso de talla 1,80…). La unión de todas las filas y columnas proporciona el conjunto de todos los valores que pueden tener las variables.

El tipo de datos que pueden representar puede ser tanto texto, números o multimedia, por ejemplo.

3.2 Selección

La investigación de detección de anomalías se ha visto frenada por falta de buenos conjuntos de datos con buenas referencias. Actualmente muchos puntos de referencia son patentados o muy artificiales.

En ODDS [11] se proporciona acceso a una gran cantidad de conjuntos de datos con anomalías de distintos dominios. Dentro de su propia distinción de conjuntos de datos se clasifican por:

-Conjuntos de datos de puntos multidimensionales
-Conjuntos de datos de gráficos de series temporales.
-Conjuntos de datos de escenarios adversos/de ataque y de seguridad
-Conjuntos de datos de gráficos de series de tiempo para la detección de eventos
-Conjuntos de datos de video de la escena de la multitud para la detección de anomalías

La elección para el trabajo es el uso de conjuntos de datos de puntos multidimensionales.
Este tipo de conjuntos este compuesto por un registro por cada punto de datos, y cada registro contiene varios atributos.

Teniendo en cuenta la revisión del buen funcionamiento del conjunto de datos y los más usados en la bibliografía, además de una diversidad entre distinto número de muestras, dimensiones y un bajo porcentaje de anomalías estos son los 3 conjuntos de datos con los que se evaluarán los algoritmos durante el trabajo:

Dataset	#muestras	#dimensiones	#anomalías (%)
Lympho	148	18	6 (4.1%)
Mammography	11183	6	260 (2.32%)
Optdigits	5216	64	150 (3%)

Tabla 3.1 Muestra de los 3 conjuntos de datos seleccionados para el trabajo.

3.2.1 Mammography dataset

El conjunto de datos originales de Mammography (Woods et al., 1993) se puso a disposición por cortesía de Aleksandar Lazarevic. Tiene 11.183 muestras. Si consideramos la exactitud de la predicción como una medida para evaluar el clasificador para este caso, la exactitud por defecto sería del 97,68% cuando cada muestra es etiquetada como no anómala. Es deseable que el clasificador prediga correctamente la mayoría de las anomalías. Para la detección de anomalías, la clase minoritaria de calcificación se considera como clase anómala y la clase sin calcificación como correcta. El conjunto de datos cuenta con 6 dimensiones y 260 datos anómalos lo que supone un 2,32% del total.

3.2.2 Optdigits dataset

Se trata de un conjunto de datos de reconocimiento óptico del depósito de aprendizaje de la UCI y es un conjunto de datos de clasificación multi clase. En el caso de los dígitos de 1-9 son datos correctos y los casos del dígito 0 son anomalías.
El conjunto de datos cuenta con 5216 puntos y 64 dimensiones en las que contiene 150 anomalías que representan un 3% del conjunto de datos.

3.2.3 Lympho dataset

El conjunto de datos de lymphografy original del depósito de la UCI es un conjunto de datos de clasificación. Es un conjunto de datos de varias clases que tiene cuatro clases, pero dos de ellas son bastante pequeñas (2 y 4 registros de datos). Por lo tanto, esas dos pequeñas clases se fusionan y se consideran anómalas en comparación con otras dos grandes clases (81 y 61 registros de datos).

El conjunto de datos cuenta con 148 puntos y 18 dimensiones en las que contiene 6 anomalías que representan un 4,1% del conjunto de datos.

3.3 Uso y manipulación de conjuntos de datos

A la hora del uso de nuestros modelos es de gran importancia el uso de los conjuntos de datos que se están usando. Es aquí donde entran en práctica los términos **overfitting** o **underfitting** conocidos en castellano como sobreajuste y subajuste.[12]

Subajuste: se dice de un modelo estadístico o un algoritmo de aprendizaje automático en el que se obtiene un bajo rendimiento cuando al no proporcionar los suficientes datos no puede captar la tendencia subyacente de los datos. El modelo probablemente realizará un gran número de predicciones erróneas.

Causas:

- Modelo demasiado simple, no es capaz de aprender relación entre datos de entrada y salida
- Falta de datos: si no contamos con los datos necesarios por completo imaginando el caso de que nuestros resultados dependan de dos variables y no contamos con una de ellas el rendimiento podría ser malo.
- Atributos relevantes insuficientes Es posible tener todos los datos pero que surja la necesidad de transformar estos para una mejor comprensión del modelo entre la relación de entrada salida.

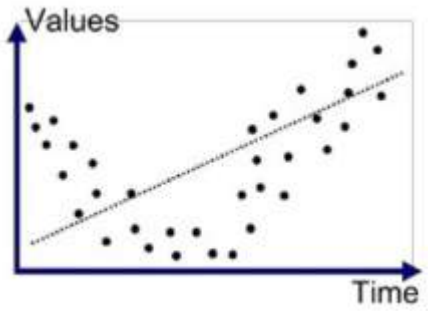

Figura 3.2 Ejemplo de modelo subajustado [13].

Sobreajuste: se dice de un modelo estadístico o un algoritmo de aprendizaje automático en el que se obtiene un bajo rendimiento cuando se entrena con un número demasiado alto de datos. Cuando un modelo se entrena con datos de más, comienza a aprender del **noise** o ruido y de las entradas de datos inexactos y no es capaz de categorizar los datos correctamente debido a demasiados detalles.

Causas:

- Modelo demasiado complejo, podrá aprender muchos de los datos de memoria.
- Los datos tienen **noise** decir como si hubiera valores atípicos y errores en los datos.

- El tamaño de los datos utilizados para los datos de entrenamiento o training data puede que no sean suficientes.

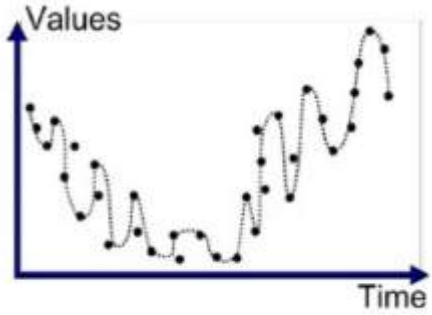

Figura 3.3 Ejemplo de modelo sobreajustado [13].

Idealmente, el caso en que el modelo hace las predicciones con error 0, se dice que tiene un buen ajuste en los datos. Esta situación es alcanzable en un punto entre el sobreajuste y el subajuste. Para entenderlo tendremos que mirar el rendimiento de nuestro modelo con el paso del tiempo, mientras que está aprendiendo del conjunto de datos de entrenamiento.

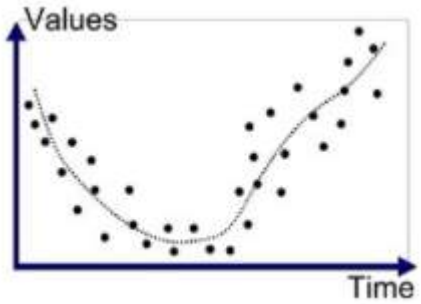

Figura 3.4 Ejemplo de modelo con un buen ajuste [13].

Con el paso del tiempo, nuestro modelo seguirá aprendiendo y por lo tanto el error del modelo en los datos de entrenamiento y pruebas seguirá disminuyendo. Si aprende durante demasiado tiempo, el modelo será más propenso a la sobrecarga debido a la presencia de ruido y de detalles menos útiles. Por lo tanto, el rendimiento de nuestro modelo disminuirá. Para conseguir un buen ajuste, nos detendremos en un punto justo antes de que el error comience a aumentar. En este punto se dice que el modelo tiene buenas habilidades en los conjuntos de datos de entrenamiento, así como en nuestro conjunto de datos de pruebas no vistas.

Es debido a esto por lo que se decide tratar nuestro conjunto de datos de la forma explicada en capítulo 3.4

3.4 Preparación del conjunto de datos

3.4.1: División entre conjunto de prueba y entrenamiento

- **Conjunto de entrenamiento**: Un subconjunto para entrenar el modelo. Este subconjunto se utilizará en la práctica para ajustar los parámetros en los algoritmos.
- **Conjunto de prueba**: Un subconjunto para probar el modelo entrenado. Este subconjunto de datos se utilizará en la práctica para proporcionar una evaluación imparcial de un último ajuste del modelo y análisis de los resultados.

Es de gran importancia que el conjunto de prueba reúna las siguientes tres condiciones [14]:

- Debe ser lo suficientemente grande como para generar resultados significativos desde el punto de vista estadístico.
- Debe ser representativo de todo el conjunto de datos. La elección del conjunto de prueba no debe tener características diferentes al del conjunto de entrenamiento. En un caso hipotético de un conjunto de datos formado por 100 muestras en las que se tienen 10 anomalías, la distribución de estas no debe ser 8 en el conjunto de prueba y 2 en el entrenamiento.
- Debe mantenerse separado de los datos de entrenamiento. Las muestras asignadas al subconjunto de prueba no deben ser introducidas en los cálculos con el subconjunto de entrenamiento. En algunas circunstancias es posible que se estén usando los datos de prueba para el entrenamiento. Esto puede perjudicar a la hora de analizar el rendimiento de los algoritmos.

En cuanto a la selección del porcentaje no hay una regla empírica que resuelva el problema. Si bien es cierto que se acepta por la comunidad [15] que, con menos datos de entrenamiento, las estimaciones de sus parámetros tienen mayor variabilidad y que con menos datos de pruebas, su estadística de rendimiento tendrá mayor varianza. En la mayoría de los casos la selección suele atribuir un porcentaje de entre 20%-30% al conjunto de prueba. En nuestro caso la selección es del 25% dejando el 75 % del conjunto de datos en el subconjunto de entrenamiento.

3.4.2 Normalización de variables de entrada

La normalización de los conjuntos de datos es un requisito común de muchos modelos de aprendizaje automático podrían comportarse mal si las características individuales no se parecen más o menos a los datos estándar distribuidos normalmente [16]: con media cero y varianza unitaria.

El objetivo de la normalización en nuestro caso es cambiar los valores de las columnas numéricas del conjunto de datos (atributos variable X) a una escala común, sin distorsionar las diferencias en los rangos de valores. Para el aprendizaje automático, no es necesario normalizar todos los conjuntos de datos. Sólo se requiere cuando las características tienen rangos diferentes. Si un atributo tiene una varianza de órdenes de magnitud mayores que otras, podría dominar la función objetivo y hacer que el estimador no pueda aprender de otras características correctamente como se espera.

En la práctica, ignoraremos la forma de la distribución y simplemente transformamos los datos para centrarlos eliminando el valor medio de cada característica, y luego los escalaremos dividiendo las características no constantes por su desviación típica [17]. Se realizará comprimiendo o extendiendo todas las variables a valores en un rango definido.

CAPÍTULO 4: Algoritmos

4.1 Introducción

Dentro de los algoritmos de aprendizaje automático hay tres enfoques principales para su clasificación. La principal diferencia entre estos tipos es el nivel de disponibilidad de los datos de la verdad sobre el dominio, que es el conocimiento previo de lo que debería ser el resultado del modelo para una entrada determinada.

El aprendizaje supervisado tiene por objetivo de aprender una función que, dada una muestra de datos y los resultados deseados, se aproxime a una función que mapee las entradas a los resultados [18].

Como técnicas principales dentro de este aprendizaje se encuentran:

- Clasificación: Un problema de clasificación es cuando la variable de salida es una categoría, como "amarillo" o "verde" o " con enfermedad" y "sin enfermedad".

- Regresión: Un problema de regresión es cuando la variable de salida es un valor real, como "euros" o "pesetas".

El aprendizaje supervisado es cuando se tienen variables de entrada (x) y una variable de salida (Y) y se utiliza un algoritmo para aprender la función de mapeo de la entrada a la salida.

$$Y = f(X)$$

El objetivo es aproximar la función de mapeo tan bien que cuando se tengan nuevos datos de entrada (x) se puedan predecir las variables de salida (Y) para esos datos.

El aprendizaje semi supervisado tiene por objetivo etiquetar los puntos de datos no etiquetados utilizando el conocimiento aprendido de un pequeño número de puntos de datos etiquetados. Al principio, cuando no tenemos ningún conocimiento, lo obtenemos de los resultados del entrenamiento. Esta configuración también utiliza conjuntos de datos de entrenamiento y prueba, donde solo los datos de entrenamiento consisten en datos normales sin anomalías. La idea es que ya se enseñó un modelo de la clase normal y que las anomalías se pueden detectar al desviarse del modelo aprendido.

El aprendizaje no supervisado no tiene (ni necesita) muestras etiquetadas, por lo que su objetivo es inferir la estructura natural presente en un conjunto de puntos de datos.

Por último, el aprendizaje no supervisado es el escogido para este trabajo y se estudia en el siguiente apartado.

4.2 Aprendizaje automático no supervisado.

El aprendizaje no supervisado es una rama importante dentro del aprendizaje automático con varias aplicaciones. Las técnicas que se engloban en el aprendizaje no supervisado no asumen que las muestras están etiquetadas (para tareas de clasificación) o tengan uno o más valores asociados a predecir (para tareas de regresión). Por tanto, no se pueden utilizar para diseñar clasificadores ni regresores; se utilizan para encontrar agrupaciones de los datos en base a uno o más criterios (por ejemplo, la distancia euclídea). Es por ello por lo que nos pueden ayudar a dividir los conjuntos de datos en dos o más agrupaciones, así como a detectar outliers (valores anómalos) [19].

En tareas de detección de anomalías las técnicas de aprendizaje no supervisado ayudan a identificar los patrones que se consideran normales. Puede que por cada patrón observado regularmente y asociado a un funcionamiento normal del sistema objeto de observación la técnica de aprendizaje no supervisado utilizada encuentre varias agrupaciones (varios clústers).
El concepto es que los enfoques de detección de anomalías no supervisadas califican los datos únicamente en función de las características naturales del conjunto de datos.

En todas las tareas del aprendizaje no supervisado deseamos aprender la estructura inherente de nuestros datos sin usar etiquetas explícitamente proporcionadas.

El aprendizaje no supervisado es muy útil en el análisis exploratorio porque puede identificar automáticamente la estructura de los datos. Por ejemplo, si un analista trata de segmentar a los consumidores, los métodos de agrupación no supervisados serían un gran punto de partida para su análisis. En situaciones en las que es imposible o poco práctico para un humano proponer tendencias en los datos, el aprendizaje no supervisado puede proporcionar conocimientos iniciales que pueden utilizarse luego para probar las hipótesis individuales.

El aprendizaje no supervisado es cuando sólo se tienen datos de entrada (X) y no hay variables de salida correspondientes.

Los problemas de aprendizaje no supervisado pueden agruparse en problemas de agrupación y asociación.

-Agrupación: Un problema de agrupación es cuando se quiere descubrir las agrupaciones inherentes en los datos, como agrupar a los clientes por el comportamiento de compra.

-Asociación: Un problema de aprendizaje de reglas de asociación es cuando se quiere descubrir reglas que describan grandes porciones de sus datos, como que las personas que compran X también tienden a comprar Y.

En muchos problemas se tienen pocos datos positivos, con lo cual se descarta la posibilidad de aplicar algoritmos supervisados ya que se necesita una cierta cantidad de casos positivos y negativos.
Hay muchos problemas de detección de anomalías que aparecen nuevos casos con el paso del tiempo y dejan de parecerse a los casos ya analizados en el pasado. Es este tipo de perfiles de algoritmos los que permiten para ambas situaciones comentadas anteriormente encontrar las anomalías.

Los algoritmos de detección de anomalías no supervisados pueden clasificarse en grandes rasgos en los siguientes cuatro grupos principales como se muestra en la Figura 4.1[20]:

1)Técnicas basadas en el vecino más cercano
2) Métodos basados en la agrupación
3) Algoritmos estadísticos
4) Técnicas subespaciales

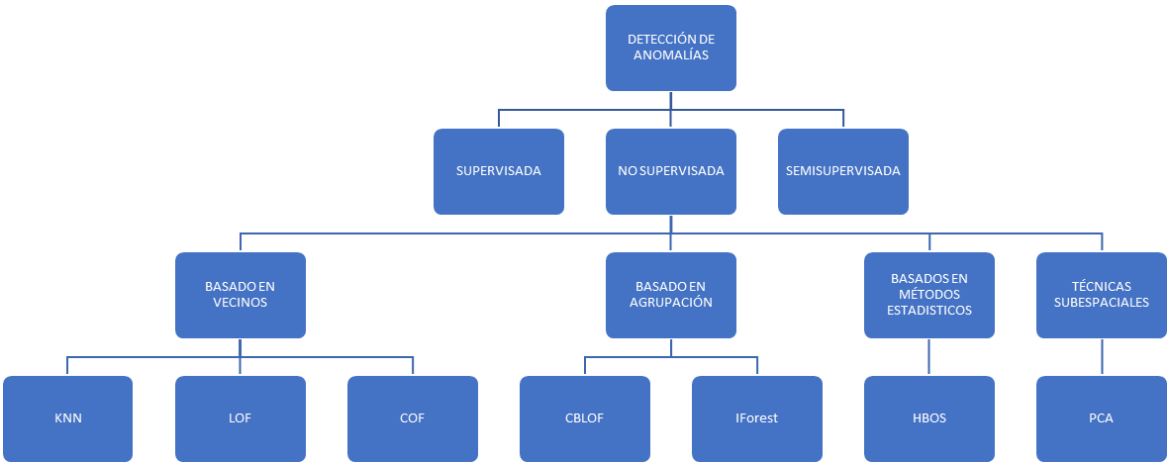

Figura 4.1 División de tipos de aprendizaje y algoritmos.

Además, existen otros algoritmos que no son miembros directos de estas categorías, a menudo basados en algoritmos de clasificación disponibles como las redes neuronales o las máquinas de soporte vectorial. La selección de este trabajo se basa en la atención prestada por la comunidad científica a este tipo de algoritmos.

4.2.1 Técnicas basadas en el vecino más cercano

4.2.1.1 k- Nearest-Neighbor(k-NN)

El algoritmo de detección de anomalías no supervisado del vecino más cercano es una forma sencilla de detectar anomalías y como su nombre indica, se centra en las anomalías globales y no es capaz de detectar anomalías locales **[21].**
.

El funcionamiento comienza por encontrar los vecinos más cercanos de cada registro en el conjunto de datos. Luego, se calcula una puntuación de anomalía usando estos vecinos, y surgen dos posibilidades:

1) Se utiliza la distancia al **kth** vecino más próximo (uno solo) denominado kth-NN.
2) Se calcula la distancia media a todos los **k-vecinos** más próximos denominado k-NN.

distancia-k (p) es igual a $d(p,q)$ donde $q \in D$ y q satisface las siguientes condiciones:

1. Para al menos k objetos $q' \in D$ se sostiene que $d(p,q') \leq d(p,q)$
2. Para la mayoría de los objetos $k-1$ $q' \in D$ se sostiene que $d(p,q') < d(p,q)$
k-vecinos(p) es el conjunto de objetos que se encuentran dentro de la hiperesfera de radio k-distancia(p) [22].

$$knn(p) = \frac{\sum_{o \in N_k(p)} d(p,o)}{|N_k(p)|}$$

Figura 4.2 Calculo del valor knn para un punto p.

Sin embargo, el valor absoluto de la puntuación depende en gran medida del conjunto de datos en sí, del número de dimensiones y de la normalización. Por lo tanto, en la práctica no es fácil seleccionar un umbral apropiado para k, pero si es necesario.

La elección del parámetro k es, por supuesto, importante para los resultados. Si se elige demasiado bajo, la estimación de la densidad para los registros podría no ser fiable. Por otra parte, si es demasiado grande, la estimación de la densidad puede ser demasiado gruesa. La mejor elección de k depende de los datos; por lo general, los valores más grandes de k reducen el efecto del ruido en la clasificación, [23] pero hacen que los límites entre las clases sean menos distintivos. Para una evaluación justa a la hora de comparar algoritmos se adjudica distintos valores de k para encontrar el mejor valor.

4.2.1.2 Local Outlier Factor (LOF)

La puntuación de la anomalía de cada muestra se llama Factor Anómalo Local.

Mide la desviación local de la densidad de una muestra dada con respecto a sus vecinos.
Se dice que esta medida es local en el sentido de que la puntuación de la anomalía depende de cuánto de aislado esté el objeto con respecto a los vecinos más cercanos **[24]**.
Más precisamente, la localidad está dada por los vecinos más cercanos, cuya distancia se utiliza para estimar la densidad local.
Comparando la densidad local de una muestra con las densidades locales de sus vecinos, se pueden identificar muestras que tienen una sustancialmente menor densidad que sus vecinos. Estos son considerados valores atípicos.

LOF consta de tres pasos de cómputo explicados a continuación:

1. Los k-vecinos más cercanos tienen que ser encontrados para cada punto p. En caso de empate de distancia del vecino k-ésimo, se usan más de k vecinos.

$$N_{k\text{-}distance(p)}(p) = \{\, q \in D \setminus \{p\} \mid d(p, q) \leq k\text{-}\ \text{distance}(p) \,\}.$$

Figura 4.3 Calculo de la distancia a los k-vecinos para p.

2. Usando estos vecinos k-nearest-neighbors Nk, la densidad local para un registro se estima calculando la densidad de accesibilidad local (LRD): mientras que dk(-) es la distancia de alcance. Excepto en algunas situaciones muy raras en cúmulos muy densos, esta es la distancia euclídea.

$$lrd_{MinPts}(p) = 1 \Big/ \left(\frac{\sum_{o \in N_{MinPts}(p)} reach\text{-}dist_{MinPts}(p, o)}{|N_{MinPts}(p)|} \right)$$

Figura 4.4 Calculo de la densidad de accesibilidad local (LRD).

3. Finalmente, la puntuación LOF de un punto p se calcula comparando el LRD de un registro con los LRD de sus k vecinos: mientras que dk(-) es la distancia de alcance. Excepto en algunas situaciones muy raras en cúmulos muy densos, esta es la distancia euclídea.

$$LOF_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{|N_{MinPts}(p)|}$$

Figura 4.5 Calculo de la puntuación LOF para un punto p.

4.2.1.3 Factor anómalo basado en la conectividad (COF)

El factor anómalo basado en la conectividad (COF) es similar a la Factor Anómalo local (LOF), pero la estimación de la densidad para los registros se realiza de manera diferente. Ambos algoritmos forman parte de técnicas basadas en el vecino más cercano, pero en este caso local**[25]**.

En LOF, los vecinos más cercanos se seleccionan en base a la distancia euclídea. Esto supone, indirectamente, que los datos se distribuyen de forma esférica alrededor de la instancia. En caso de violar la suposición, por ejemplo, si las características tienen una correlación lineal directa, la estimación de la densidad es incorrecta. El COF quiere compensar esta deficiencia y estima la densidad local de los vecinos utilizando un enfoque de camino más corto, llamado distancia de encadenamiento. Matemáticamente, esta distancia de encadenamiento es el mínimo de la suma de todas las distancias que conectan todos los vecinos k y la instancia. Para ejemplos simples, donde las características están obviamente correlacionadas, este enfoque de estimación de densidad es mucho más preciso.

Para empezar, se añade la instancia más cercana a la instancia dada al conjunto de vecinos. La siguiente instancia añadida es la que tiene la distancia mínima entre el conjunto de vecinos existente y el resto de los conjuntos de vecinos. La distancia entre una instancia y un conjunto de instancias se define como la distancia mínima entre la instancia dada y cualquier instancia perteneciente al conjunto dado. De esta forma el conjunto de vecinos crece hasta que alcanza el tamaño k.. Una vez calculado el conjunto de vecinos, la puntuación atípica (COF) se calcula de la misma manera que la LOF.

Sea p ∈ D y sea k un número entero positivo. La conectividad basada en un atípico mediante el factor (COF) en p con respecto a sus k vecinos se define de acuerdo con [26] como:

$$COF_k(p) = \frac{|N_k P|.ac-dist_{n_{k(p)}}(p)}{\sum_{o \in n_k(p)} ac-dist_{n_{k(o)}}(o)}$$

Figura 4.6 Calculo de la puntuación COF para un punto p.

4.2.1.4 Correlación local integral (LOCI)

Se realiza una explicación teórica sobre este algoritmo pese a no ser seleccionado para la fase de ejecución ni análisis teniendo en cuenta diversas referencias en cuanto a los tiempos de cómputo.

Para todos los algoritmos anteriores, elegir k es una decisión crucial para el rendimiento de la detección. No obstante, el algoritmo de la integral de correlación local (LOCI) aborda esta cuestión utilizando un enfoque de maximización **[27].**
. La idea básica es que se utilicen todos los valores posibles de k para cada registro y finalmente se tome la puntuación máxima. Para lograr este objetivo, el LOCI define el grupo de vecinos r utilizando un radio r, que se va expandiendo con el tiempo. Además, la estimación de la densidad local es diferente en el LOCI: Compara dos grupos de vecinos de diferente tamaño en lugar de la proporción de las densidades locales. Un parámetro α controla la proporción de los diferentes grupos de vecinos.
En la práctica esta solución para eliminar el parámetro crítico k tiene un precio [28]. Típicamente, los algoritmos de detección de anomalías basados en el vecino más cercano tienen una complejidad computacional de $O(n^2)$ para encontrar los vecinos más cercanos. En el caso de LOCI, la complejidad aumenta a $O(n^3)$, lo que hace que LOCI sea demasiado lento para conjuntos de datos más grandes. Es este el motivo de su exclusión para la comparación de algoritmos basados en la distancia a los vecinos.

Los tiempos de cómputo en la fase de ejecución suponían para el mayor conjunto de datos tiempos superiores a 3 días.

4.2.2 Métodos basados en la agrupación

4.2.2.1 Factor atípico local basado en grupos (CBLOF)

Todos los algoritmos anteriores de detección de anomalías se basan en la estimación de la densidad utilizando los vecinos más cercanos. Por el contrario, el factor de valores atípicos locales basados en **clusters** o grupos (CBLOF) utiliza la agrupación para determinar áreas densas en los datos y realiza una estimación de la densidad para cada grupo posteriormente [29].

En la práctica, el usuario tiene la opción de seleccionar la partición usando dos parámetros α y β que serán valores de coeficiente para decidir los grupos pequeños y grandes.

Después de la agrupación, CBLOF utiliza una heurística para clasificar los grupos resultantes en grupos grandes y pequeños. Finalmente, se calcula una puntuación de anomalía por la distancia de cada instancia a su centro de cúmulo multiplicada por las instancias que pertenecen a su cúmulo. Para los cúmulos pequeños, se utiliza la distancia al cúmulo grande más cercano. El procedimiento de utilizar la cantidad de miembros del cúmulo como factor de escala debe estimar la densidad local de los cúmulos, tal y como han indicado los autores.

Las puntuaciones CBLOF se calculan mediante la siguiente formula [30]:

$$CBLOF(p) = \begin{cases} |C_i| \cdot \min(d(p, C_j)) \text{ if } C_i \in SC \text{ where } p \in C_i \text{ and } C_j \in LC \\ |C_i| \cdot d(p, C_i) \text{ if } C_i \in LC \text{ where } p \in C_i \end{cases}$$

Figura 4.7 Calculo de la puntuación CBLOF para un punto p.

Asignan una puntuación de anomalía basándose en la distancia al grupo grande más cercano multiplicado por el tamaño del grupo al que pertenece el objeto.

4.2.2.2 Isolation Forest (IForest)

El Bosque de Aislamiento 'aísla' las observaciones seleccionando al azar una característica y luego seleccionar al azar un valor de división entre el máximo y mínimo valores mínimos de la característica seleccionada **[31]**.

Dado que la partición recursiva puede ser representada por una estructura de árbol, el número de divisiones necesarias para aislar una muestra es equivalente a la longitud del camino desde el nodo raíz hasta el último nodo.

Esta longitud de la ruta, promediada sobre un bosque de árboles tan aleatorios, es una medida de la normalidad y nuestra función de decisión.
La división aleatoria produce caminos notablemente más cortos para las anomalías.
Por lo tanto, cuando un bosque de árboles al azar produce colectivamente entre los nodos longitudes de camino más cortas para muestras particulares, es muy probable que sean anomalías.
La idea de identificar una observación normal frente a una anormal puede observarse en un punto normal, requiere que se identifiquen más particiones que una anomalía.

Al igual que con otros métodos de detección de anomalías, se requiere un punto anómalo para la toma de decisiones. En el caso del Bosque de Aislamiento, se define como:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}}$$

Figura 4.8 Cálculo de anomalía para Iforest.

donde h(x) es la longitud del camino de observación x, c(n) es la longitud media del camino de búsqueda fallida en un Árbol de Búsqueda Binario y n es el número de nodos externos. Se puede leer más sobre la puntuación de la anomalía y sus componentes en [31].
A cada observación se le da una puntuación de la anomalía y se puede tomar la siguiente decisión en base a ella:

- Una puntuación cercana a 1 indica anomalías.
- Una puntuación mucho menor que 0,5 indica observaciones normales.
- Si todas las puntuaciones se acercan a 0,5, entonces la muestra completa no parece tener anomalías claramente diferenciadas.

4.2.3 Algoritmos estadísticos

4.2.3.1 Histogram-based Outlier Score (HBOS)

La idea básica de la detección de anomalías basada en histogramas es que para cada característica del conjunto de datos se crea un histograma **[32].** Luego, para cada instancia del conjunto de datos, se multiplica la altura inversa de los recipientes en que reside (que representan la estimación de la densidad) de todas las características. A primera vista, podría parecer un poco contraproducente que se descuiden las dependencias entre las características. Sin embargo, proporciona una gran ventaja a la velocidad de procesamiento.

El parámetro crítico para el correcto funcionamiento es el número de bins (contenedores) k. Una regla de uso frecuente para la obtención del valor k está fijando k a la raíz cuadrada del número de instancias N [33].

Para cada característica individual (dimensión), primero se construye un histograma de variable única. Si la característica está comprendida por datos categóricos, se realiza el recuento de los valores de cada categoría y se computa la frecuencia relativa (altura del histograma).
Los valores sucesivos se agrupan en un solo recipiente donde N es el número total de instancias y k el número de contenedores. Ya que el área de un recipiente en un histograma representa el número de observaciones, es el mismo para todos los contenedores en nuestro caso.
Debido a que el ancho del contenedor está denotado por el primer y último valor, y es el misma para todos los contenedores, la altura de cada contenedor individual puede ser calculada.
Esto significa que los contenedores que cubren un intervalo mayor del rango de valores tienen menos altura y representar así una menor densidad.

Ahora, para cada dimensión d, se ha calculado un histograma individual donde la altura de cada uno de los contenedores representa una estimación de la densidad.
El binomio simple representa una estimación de la densidad. Los histogramas se normalizan de tal manera que la altura máxima es de 1.0. Esto asegura un peso igual de cada rasgo para la anomalía. Finalmente, el HBOS de cada instancia p se calcula utilizando la altura correspondiente de los contenedores donde se encuentra la instancia:

$$HBOS(p) = \sum_{i=0}^{d} log(\frac{1}{hist_i(p)})$$

Figura 4.9 Calculo de la puntuación HBOS para un punto p.

La puntuación es una multiplicación del inverso de las densidades estimadas asumiendo independencia de las características similares. Esto también podría ser visto como el inverso de un modelo discreto de probabilidad de Bayes. En lugar de la multiplicación, tomamos la suma de los logaritmos que es básicamente la misma:

$$(\log(a * b) = \log(a) + \log(b))$$

y la aplicación de log $(*)$ no cambia el orden de las puntuaciones.

4.2.4 Técnicas subespaciales

4.2.4.1 Principal Component Analysis (PCA)

El análisis de componentes principales es una técnica comúnmente utilizada para detectar subespacios en los conjuntos de datos **[34].** En este caso se va a usar como técnica de detección de anomalías, de manera que las desviaciones de los subespacios normales pueden indicar instancias anómalas. Los componentes principales son los vectores propios de la matriz de covarianzas-covarianzas y, por lo tanto, su cálculo contiene cierta dificultad: las anomalías tienen una gran influencia en la matriz de covarianzas-covarianzas y la estimación de la densidad podría ser incorrecta. Una vez que se determinan los componentes principales, la cuestión es qué componentes deben utilizarse para puntuar las instancias anómalas. El uso de los componentes principales muestra desviaciones globales de la mayoría de los datos, mientras que el uso de los componentes menores puede indicar desviaciones locales más pequeñas.

Se explica el modelo con brevedad [35]:

Sea la matriz estandarizada **Xs**. Utilizando la descomposición de valores singulares (SVD), PCA transforma la matriz de datos **Xs** en una nueva matriz $\mathbf{T} = [t_1\ t_2\ \cdots\ t_m] \in \mathbb{R}^{n\times m}$ de una variable no correlacionada llamada puntuación o componentes principales.

Cada nueva variable es una combinación lineal de las variables originales, de manera que **T** se obtiene de **Xs** a partir de transformaciones ortogonales (rotaciones) diseñadas por $\mathbf{P} = [p_1\ p_2\ \cdots\ p_m] \in \mathbb{R}^{m\times m}$, los vectores de la columna $p_i \in R^m$ de la matriz $\mathbf{P} \in R^{m\times m}$ (también conocidos como vectores de carga) están formados por los vectores propios asociados a la matriz de varianzas-covarianzas de **Xs**, es decir, ∑. La matriz de varianzas-covarianzas, ∑, se define como sigue:

$$\Sigma = \frac{1}{n-1}\mathbf{X}_s^T\mathbf{X}_s = P\Lambda P^T \quad \text{with} \quad PP^T = P^TP = I_n,$$

Donde **In** es la matriz de identidad en R^{nxn} [36].

La precisión del modelo de PCA depende de una buena elección de cuántos componentes principales (PCs) se conservan [37]. Si el número de PC es l, no se estima correctamente, el modelo puede subestimar (subestimación) o sobreestimar (sobreestimación) los datos.Al sobreestimar el número de PCs se puede introducir ruido y el modelo no logrará captar parte de la información. En el caso de subestimar el número de PCs es posible que no se capturen características importantes de los datos, y esto conlleva una menor calidad del modelo.

4.3 Métricas para el análisis

Se enumeran aquí las métricas seleccionadas para la evaluación y comparación de los algoritmos. Estas métricas se basan principalmente sobre las etiquetas de anomalías/esperanzas booleanas asignadas a un punto de datos determinado.
Sin embargo, al proporcionar una salida, los algoritmos pueden no proporcionar una respuesta booleana: en cambio, suelen proporcionar una anomalía numérica que indica cuán anómalo es un punto de datos en relación con los otros.
Para el trabajo no tenemos en cuenta las métricas relacionadas con tiempo de detección ni tiempo de detección de las anomalías, ya que normalmente depende de los recursos hardware disponibles [38].

4.3.1 ROC-AUC

Una curva ROC (curva de característica operativa del receptor) es una gráfica que muestra el rendimiento de un modelo de clasificación en todos los umbrales de clasificación. Esta curva representa dos parámetros:

La curva ROC-AUC es una medida de rendimiento para el problema de clasificación en varios umbrales. ROC es una curva de probabilidad y AUC representa el grado o medida de separabilidad. Será a través de la métrica AUC (Area Under the ROC Curve) que representa un valor del área que queda por debajo de la curva ROC,la encargada de comparar unos modelos con otros indicando cuánto es capaz el modelo de distinguir entre clases. Cuanto más alto es el AUC, mejor es el modelo para predecir verdaderos como verdades y falsos como falsos.

La curva ROC se traza con la Sensibilidad [4.1] frente la Especificidad [4.2] donde la Sensibilidad está en el eje Y, y el eje X está compuesto por 1- Especificidad:

La evaluación de esta medida se clasifica de la siguiente forma [39]:

Valor cerca de 1: El modelo es excelente, tiene una gran capacidad de separabilidad. En este caso es perfectamente capaz de distinguir entre la clase positiva y la clase negativa.

Valor cerca de 0,5: El modelo no tiene ninguna capacidad de separación de clases. Es la peor situación y el modelo no tiene capacidad de discriminación para distinguir entre clase positiva y negativa.

4.3.2 Matriz de confusión

La matriz de confusión es una herramienta que nos muestra el desempeño de un algoritmo de clasificación, describiendo cómo se distribuyen los valores reales y nuestras predicciones [40] mediante 4 distintos casos basados en 4 variables que se comentan a continuación:

Valores actuales: Verdadero o Falso.

Valores predichos: Positivo o Negativo

		Predicción	
		Positivos	**Negativos**
Observación	**Positivos**	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (VN)

Figura 4.10 Matriz de confusión [41].

A partir de la matriz anterior tenemos cuatro clasificaciones posibles:

Verdadero positivo (VP): Se predice que es positivo y es verdad, fue clasificado correctamente.
Verdadero negativo (VN): Se predice que es negativo y es verdad, fue clasificado correctamente.
Falso positivo (FP): Se predice que es positivo y es falso, fue clasificado incorrectamente.
Falso negativo (FN): Se predice que es negativo y es falso, fue clasificado incorrectamente.

4.3.2.1 Sensibilidad

En ingles se conoce como “Recall”, “Sensibility” y como Tasa de Verdaderos Positivos (True Positive Rate) o TPR [42]. Es la proporción de casos positivos que fueron correctamente identificadas por el algoritmo.

Se calcula mediante la siguiente ecuación:

$$TPR = \frac{VP}{FN+VP}$$

En la práctica, usaremos esta medida a partiz del calculo de la matriz de confusión, y también mediante la llamada a un método que calcula la sensibilidad e incluye como parámetro “average”. Este parámetro es necesario para los objetivos multiclase/multi etiqueta. Asignando el valor ‘macro’ al parámetro *average* se calculan las métricas de cada etiqueta, y encuentra su media no ponderada. Por lo tanto, no tiene en cuenta el desequilibrio de las etiquetas y se obtendrá una sensibilidad distinta a la obtenida mediante la matriz de confusión

4.3.2.2 Especificidad

En ingles conocido como “Especificity” y también conocido como Tasa de Verdaderos Negativos, (“true negative rate”) o TNR [42]. Se trata de los casos negativos que el algoritmo ha clasificado correctamente. Expresa cuan bien puede el modelo detectar esa clase.

Se calcula mediante la siguiente ecuación:

$$TNR = \frac{VN}{VN+FP}$$

Junto con la sensibilidad, ambas medidas indican la capacidad de discriminar los casos positivos de los negativos.

4.3.2.3 Precisión

En ingles conocido como (“Accuracy”) se refiere a la dispersión del conjunto de valores obtenidos a partir de mediciones repetidas de una cierta magnitud. Cuanto menor es la dispersión mayor la precisión. Se representa por la proporción entre el número de predicciones correctas (tanto positivas como negativas) y el total de predicciones [43].La mejor precisión viene dada por el valor 1 mientras que la peor precisión viene dada por el valor 0.

En forma práctica la precisión es la cantidad de predicciones positivas que fueron correctas.

$$AC= \frac{VP+VN}{VN+FP+FN+VP}$$

4.3.2.4 Exactitud

En ingles conocido como (“Precision”) . Se refiere a lo cerca que está el resultado de una medición del valor verdadero. Se representa por la proporción entre los positivos reales predichos por el algoritmo y todos los casos positivos. Es conocido por Verdadero Positivo (o “True positive rate”).

En forma práctica es el porcentaje de casos positivos detectados. Se calcula a partir de la siguiente formula:

$$P= \frac{VP}{FP+VP}$$

4.3.2.5 F1

Combina precisión y sensibilidad en una sola métrica. (Precision y Recall en una sóla metrica) por ello es de gran utilidad cuando la distribución de las clases es desigual.

$$F1= 2 * \frac{SENSIBILIDAD*PRECISIÓN}{SENSIBILIDAD+PRECISIÓN}$$

Conforme a estas métricas tenemos cuatro casos posibles para cada clase:

Alta exactitud y sensibilidad: el modelo maneja perfectamente esa clase.
Alta exactitud y baja sensibilidad: el modelo no detecta la clase muy bien, pero cuando lo hace es altamente confiable.
Baja exactitud y alta sensibilidad: El modelo detecta bien la clase, pero también incluye muestras de otras clases.
Baja exactitud y sensibilidad: El modelo no logra clasificar la clase correctamente.

Maximizar el valor de F1, conlleva a un mejor rendimiento del algoritmo [44].

4.3.3 Kappa

El Coeficiente kappa de Cohen mide la concordancia entre dos examinadores en sus correspondientes clasificaciones de N elementos en C categorías mutuamente excluyentes. Es una medida cuantitativa de fiabilidad para dos calificadores que están calificando la misma cosa, corregida por la frecuencia con que los calificadores pueden coincidir por casualidad.
La ecuación para κ es [45]:

$$\kappa = \frac{\Pr(a) - \Pr(e)}{1 - \Pr(e)}$$

Figura 4.11 Calculo de coeficiente Kappa.

donde Pr(a) es el acuerdo observado relativo entre los observadores, y Pr(e) es la probabilidad hipotética de acuerdo por azar, utilizando los datos observados para calcular las probabilidades de que cada observador clasifique aleatoriamente cada categoría.
A partir de ahí si κ = 1 significa que los evaluadores están de acuerdo y κ = 0 cuando no hay acuerdo entre los calificadores.

El valor de kappa puede variar entre 0-1. Una puntuación de 0 significa que hay un acuerdo aleatorio entre los clasificadores, mientras que una puntuación de 1 significa que hay un acuerdo completo entre los clasificadores.
Es posible que el resultado de la métrica sea negativo [46] lo que implica que no hay un acuerdo efectivo entre los dos calificadores o que el acuerdo es peor que el azar.

4.4 Herramientas

4.4.1 Python

Es un lenguaje de programación de alto nivel, interpretado y orientado a objetos, con semántica dinámica **[47]**.

La selección del lenguaje viene dada por una serie de características que se comentan a continuación, cruciales para la práctica de algoritmos de aprendizaje automático [48].

Ofrece buen rendimiento desde desarrollo hasta el despliegue y mantenimiento. Facilita los proyectos debido a su simplicidad, consistencia y flexibilidad, así como acceso a grandes bibliotecas y entornos de trabajo para el aprendizaje automático.
En cuanto al código es conciso y legible a la hora de implementar algoritmos. Es un lenguaje a su vez intuitivo y fácil de aprender.
Consta de una gran comunidad que facilita tanto el aprendizaje como la resolución de problemas durante el trabajo.

4.4.2 Anaconda

Distribución de Python con herramientas de instalación y gestión de paquetes **[49]**.
Gestor de entornos que proporciona la posibilidad de crear diferentes entornos de Python, cada uno con su propia configuración [50].

Dentro de Anaconda se usa Conda como gestor de paquetes facilitando la búsqueda e instalación de paquetes. Conda ayuda a organizar todas estas herramientas. Aunque Anaconda viene con muchas de ellas listas para usar, a veces necesitarán ser cambiadas. Conda es como el asistente que hace un balance de todas las herramientas.

4.4.3 PyOD

Es una librería completa y escalable de Python diseñada para detección de anomalías en datos multivariados. Proporciona acceso a más de 20 algoritmos como se muestra en la Figura 5 de detección de anomalías bajo una API (Una interfaz de programa de aplicación (API) es un conjunto de rutinas, protocolos y herramientas para construir aplicaciones de software) única y bien documentada. Recoge una amplia gama de técnicas que van desde el aprendizaje supervisado a las técnicas de aprendizaje no supervisado **[51]**.
Su código es abierto con documentación detallada y ejemplos de varios algoritmos.

Type	Abbr	Algorithm
Linear Model	PCA	Principal Component Analysis (the sum of weighted projected distances to the eigenvector hyperplanes)
Linear Model	MCD	Minimum Covariance Determinant (use the mahalanobis distances as the outlier scores)
Linear Model	OCSVM	One-Class Support Vector Machines
Proximity-Based	LOF	Local Outlier Factor
Proximity-Based	COF	Connectivity-Based Outlier Factor
Proximity-Based	CBLOF	Clustering-Based Local Outlier Factor
Proximity-Based	LOCI	LOCI: Fast outlier detection using the local correlation integral
Proximity-Based	HBOS	Histogram-based Outlier Score
Proximity-Based	kNN	k Nearest Neighbors (use the distance to the kth nearest neighbor as the outlier score
Proximity-Based	AvgKNN	Average kNN (use the average distance to k nearest neighbors as the outlier score)
Proximity-Based	MedKNN	Median kNN (use the median distance to k nearest neighbors as the outlier score)
Proximity-Based	SOD	Subspace Outlier Detection
Probabilistic	ABOD	Angle-Based Outlier Detection
Probabilistic	FastABOD	Fast Angle-Based Outlier Detection using approximation
Probabilistic	SOS	Stochastic Outlier Selection
Outlier Ensembles	IForest	Isolation Forest
Outlier Ensembles		Feature Bagging
Outlier Ensembles	LSCP	LSCP: Locally Selective Combination of Parallel Outlier Ensembles
Outlier Ensembles	XGBOD	Extreme Boosting Based Outlier Detection **(Supervised)**
Neural Networks	AutoEncoder	Fully connected AutoEncoder (use reconstruction error as the outlier score)
Neural Networks	SO_GAAL	Single-Objective Generative Adversarial Active Learning
Neural Networks	MO_GAAL	Multiple-Objective Generative Adversarial Active Learning
Outlier Ensembles		Feature Bagging
Outlier Ensembles	LSCP	LSCP: Locally Selective Combination of Parallel Outlier Ensembles
Combination	Average	Simple combination by averaging the scores
Combination	Weighted Average	Simple combination by averaging the scores with detector weights
Combination	Maximization	Simple combination by taking the maximum scores
Combination	AOM	Average of Maximum
Combination	MOA	Maximum of Average

Figura 4.12 Algoritmos PyOD [52].

4.4.4 Google colab

Google Colaboratory es un entorno gratuito de Jupyter Notebook que no requiere configuración y que se ejecuta completamente en la nube **[53]**.

Mediante Colab, se puede importar un conjunto de datos, entrenar un clasificador con dicho conjunto de datos y evaluar el modelo a través del navegador. Con tan solo usar los cuadernos de Colab, el código se ejecutará en los servidores en la nube de Google, lo que permite aprovechar la potencia del hardware de Google, incluidas GPU y CPU, independientemente de la potencia del equipo.

Durante la ejecución del código en nuestro caso el ordenador personal no era capaz de realizar el computo en los tiempos esperados con los grandes conjuntos de datos. Mediante el uso de esta herramienta el código se ejecutó con un notable aumento de velocidad.

4.4.5 Numpy

Su uso es principalmente para los conjuntos de datos. Sus conceptos de vectorización, indexación y difusión de son los estándares de cálculo de la informática de matrices hoy en día debido a su rapidez y versatilidad **[54]**.

NumPy trae el poder de cómputo de lenguajes como C y Fortran a Python.
NumPy forma la base de potentes bibliotecas de aprendizaje automático como scikit-learn y SciPy. A medida que el aprendizaje automático crece, también lo hace la lista de bibliotecas construidas en NumPy [55].

4.4.6 Numba

Es un compilador basado en LLVM Python JIT **[56]**. Se centra en la computación científica y orientada a los arreglos. Empezando por la sintaxis simple de Python, Numba [57] compila un subconjunto de la en un código de máquina eficiente que es comparable en a un lenguaje tradicional compilado.

4.4.7 Scipy

Es un ecosistema basado en Python de software de código abierto para matemáticas, ciencia e ingeniería **[58]**. En particular, contiene Numpy, paquete comentado anteriormente y su propia librería Scipy que contiene una colección de algoritmos numéricos para optimización, estadísticas y mucho más dependiendo del dominio.

4.4.8 Scikit-learn

Es una biblioteca para aprendizaje automático de software libre para el lenguaje de programación Python **[59]**. Incluye varios algoritmos de clasificación, regresión y análisis de grupos entre los cuales están máquinas de vectores de soporte, bosques aleatorios, K-means y DBSCAN. Está diseñada para interoperar con las bibliotecas numéricas y científicas NumPy y SciPy.

El uso de esta biblioteca es de gran importancia para el código ya que forma parte desde la preparación de los datos hasta las métricas de evaluación de resultados de los algoritmos.

CAPITULO 5: Ejecución

Para la simplificación de las distintas ejecuciones del código y como recurso para la comprensión de las selecciones de valores de parámetros realizadas durante la fase de ejecución, se muestra a continuación una figura que representa el flujo de trabajo seguido para la obtención de resultados.

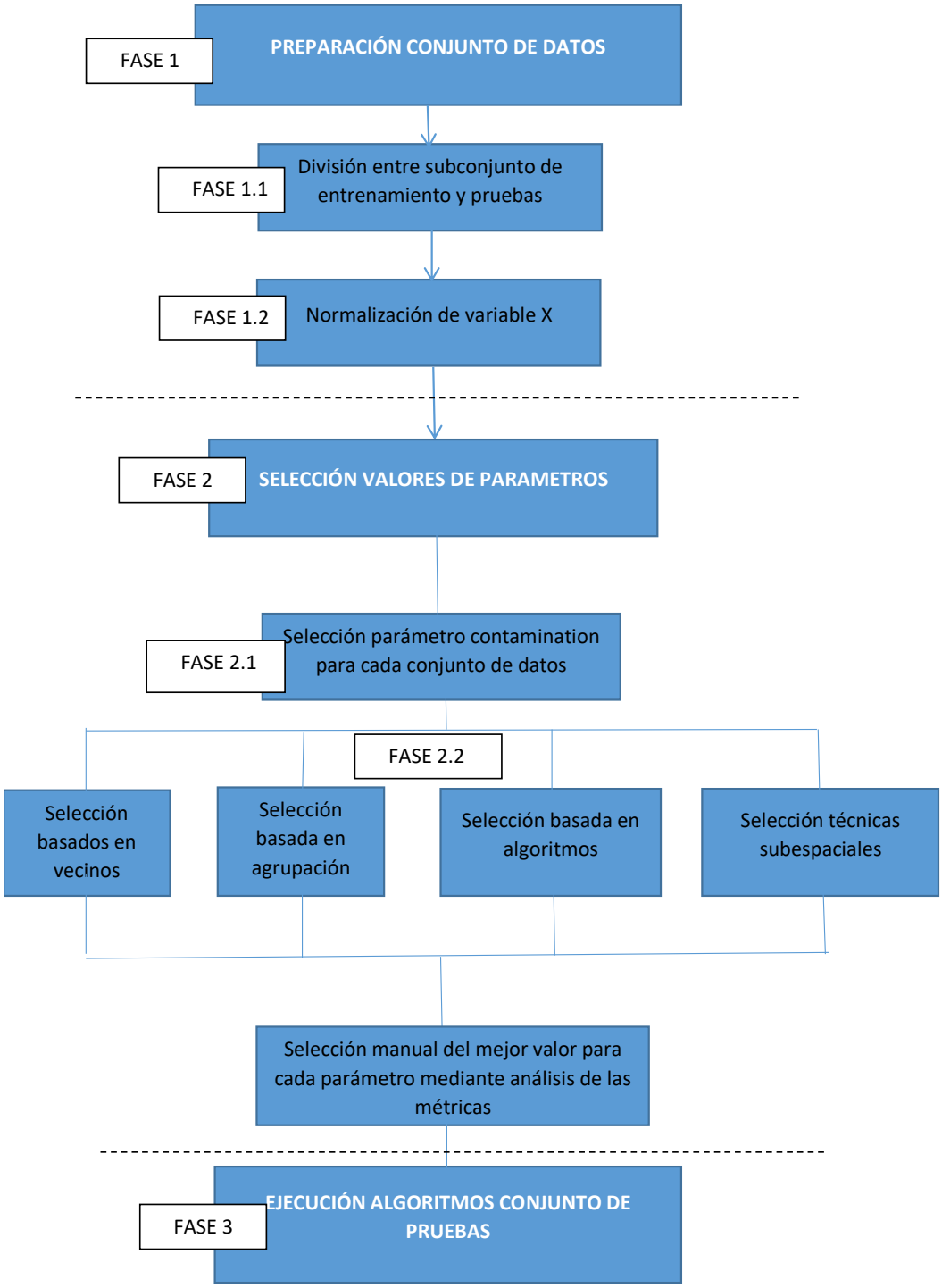

Figura 5.1 Flujo de trabajo ejecución.

5.1 Preparación de los conjuntos de datos

La preparación de los conjuntos de datos queda representada en la Figura 5.1 en la Fase 1.

Tras la importación del conjunto de datos al código, y la posterior selección de las variables X e y se procede a la división entre el conjunto de entrenamiento y pruebas.

Para los 3 conjuntos de datos se realiza una separación del conjunto de datos en 2 partes comentadas teóricamente y mediante el método:

sklearn.model_selection.train_test_split(*arrays, **options)

Obteniendo como resultado a la llamada train_test_split(X,y,test_size=0.25,random_state=42)

4 variables denominadas:

X_train,X_test,y_train,y_test.

Esta llamada queda reflejada en la Fase 1.1

Se selecciona un porcentaje de 25% para el subconjunto de pruebas y 75% para el de entrenamiento y es de importancia el parámetro *random_state* para controlar que la forma de separar el conjunto de datos se reproduzca idénticamente en todas las llamadas y por lo tanto la división será exactamente igual en cada ejecución del código. Se asigna el valor 42 por su popularidad y su interés para la ciencia [60]pese a poder elegirse cualquier valor cualquier número entero.

Una vez contamos con las 4 variables, se procede a normalizar la variable X tanto para el subconjunto de entrenamiento como el de prueba.

Esta normalización se realiza mediante la llamada al método

pyod.utils.utility.standardizer(X, X_t=None, keep_scalar=False)

en la Fase 1.2 en el flujo de trabajo.

Se lleva a cabo la normalización Z de los datos para que las muestras de entrada tengan media cero y varianza unitaria.

Una vez tenemos el conjunto dividido y normalizado se procede a la siguiente fase.

5.2 Selección parámetro común *contamination*

La Fase 2 de la ejecución conlleva la realización de pruebas de asignación de distintos valores. Esta fase está dividida en dos sub-fases:

- Fase 2.1 que conlleva la selección del mejor valor para el parámetro *contamination*, con distintas ejecuciones y una selección manual posterior del mejor valor para cada conjunto de datos.
- Fase 2.2 del mismo modo que para el parámetro *contamination* pero esta vez el procedimiento se realiza para un parámetro de cada algoritmo en cada conjunto de datos.

Para la buena evaluación de los algoritmos se realizan diversas pruebas en los distintos algoritmos y sus parámetros.

Todos los algoritmos de la librería PyOD cuentan con un parámetro común denominado: "*contamination*".

Este parámetro se utiliza a la hora de realizar el ajuste para definir el umbral en la función de decisión. Esto conlleva a que independientemente del algoritmo utilizado, debido al conocimiento previo de número de anomalías en el conjunto de datos se debe indicar la proporción de contaminación para un mejor rendimiento del algoritmo. En el caso práctico para la primera ejecución de cualquier algoritmo realizaremos la llamada mediante la modificación del parámetro, variando sus valores y seleccionando el resultado. Este parámetro es un *float* y acepta valores entre (0.,0.5) teniendo por defecto el valor 0.1.

Se realizan diversas ejecuciones de un bucle variando el valor del parámetro *contamination* para obtener el número de muestras normales y anomalías conocido previamente.

Independientemente del algoritmo, y sus parámetros, este array viene dado por la división del conjunto de datos por lo que para la optimización debemos tratar mediante el parámetro obtener valores similares. En la práctica mediante un contador de 0 (datos normales) y 1 (datos anómalos) se trata de obtener el mismo número con la variación del parámetro: Fase 2.1.

Esto conlleva la siguiente selección para cada conjunto de datos:

Lympho:

Con un total de 148 muestras, se asignarán 111 muestras para el conjunto de entrenamiento.
De estas muestras se identifican 5 anomalías y 106 datos normales:

LYMPHO				
Aproximación 1 Valor contamination:	0.1	0.2	0.3	0.4
	{0: 100, 1: 11}	{0: 89, 1: 22}	{0: 78, 1: 33}	{0: 67, 1: 44}
Aproximación 2 Valor contamination:	0.01	0.02	0.03	**0.04**
	{0: 109, 1: 2}	{0: 108, 1: 3}	{0: 107, 1: 4}	{0: 106, 1: 5}

Tabla 5.1 Valor de parámetro *contamination* para Lympho.

Obtenemos como valor optimo 0.04.

Optdigits:

Con un total de 5216 muestras, para el conjunto de entrenamiento contaremos con 3912 muestras.
De estas muestras se identifican 108 anomalías y 3804 datos normales.

OPTDIGITS				
Aproximación 1				
Valor contamination:	0.1	0.2	0.3	0.4
	{0: 3520, 1: 392}	{0: 3129, 1: 783}	{0: 2738, 1: 1174}	{0: 2347, 1: 1565}
Aproximación 2				
Valor contamination:	0.01	0.02	0.03	0.04
	{0: 3872, 1: 40}	{0: 3833, 1: 79}	{0: 3794, 1: 118}	{0: 3755, 1: 157}
Aproximación 3				
Valor contamination:	0.025	0.026	0.027	0.028
	{0: 3810, 1: 102}	{0: 3806, 1: 106}	{0: 3802, 1: 110}	{0: 3716, 1: 196}
Aproximación 4				
Valor contamination:	0.0271	0.0272	0.0273	**0.0274**
	{0: 3806, 1: 106}	{0: 3805, 1: 107}	{0: 3805, 1: 107}	{0: 3804, 1: 108}

Tabla 5.2 Valor de parámetro *contamination* para Optdigits.

Obtenemos como valor optimo 0.0274

Mammography

Con un total de 11183 muestras, para el conjunto de entrenamiento contaremos con 8387 muestras.
De estas muestras se identifican 197 anomalías y 8190 datos normales.

MAMMOGRAPHY				
Aproximación 1 Valor contamination:	0.1	0.2	0.3	0.4
	{0: 7548, 1: 839}	{0: 6709, 1: 1678}	{0: 5871, 1: 2516}	{0: 5377, 1: 3010}
Aproximación 2 Valor contamination:	0.01	0.02	0.03	0.04
	{0: 8303, 1: 84}	{0: 8219, 1: 168}	{0: 8135, 1: 252}	{0: 8051, 1: 336}
Aproximación 3 Valor contamination:	0.021	0.022	0.023	0.024
	{0: 8210, 1: 177}	{0: 8202, 1: 185}	{0: 8194, 1: 193}	{0: 8185, 1: 202}
Aproximación 4 Valor contamination:	0.0233	**0.0234**	0.0235	0.0236
	{0: 8191, 1: 196}	{0: 8190, 1: 197}	{0: 8190, 1: 197}	{0: 8189, 1: 198}

Tabla 5.3 Valor de parámetro *contamination* para Mammography.

Obtenemos como valor optimo 0.0234

Para la continuación del trabajo se asignarán estos valores para el parámetro *contamination* en cada una de las ejecuciones.

5.3 Selección valor del parámetro independiente del algoritmo

Selección parámetros del algoritmo: Fase 2.2.

La selección del valor óptimo para los parámetros modificados se realiza en base a las métricas Sensibilidad, Especificidad y Exactitud. De las diversas ejecuciones con distintos valores se selecciona el valor que de un resultado más próximo a 1 en estas tres métricas. En caso de que los resultados obtenidos para distintos valores del parámetro coincidan, se seleccionará el valor que de un número más próximo a 1 en la métrica ROC-AUC.

La asignación de valores al parámetro se realiza mediante un bucle. Para cada algoritmo y conjunto de datos tras una primera ejecución del código se selecciona manualmente el valor que proporciona los mejores resultados y se realiza una segunda ejecución adaptando los valores al que ha resultado en la primera ejecución ser el mejor valor. Para la claridad de este apartado se incluye únicamente la segunda ejecución del código solo si alguno de los nuevos valores obtiene mejores resultados.

La columna del valor con el mejor resultado se marca en rojo.

Durante esta fase debido a los largos tiempos de ejecución (más de 24 horas) en el ordenador personal se usa la herramienta Colab en lugar de la ejecución mediante Ipython, herramienta seleccionada en el inicio para la ejecución del código.

Los algoritmos basados en vecinos más cercanos comparten el parámetro k: que indica el número de vecinos a seleccionar para realizar los cálculos. Los algoritmos KNN, LOF y COF comparten el parámetro k que indica el número de vecinos a seleccionar para realizar los cálculos.

5.3.1 KNN

LYMPHO							
n_neighbors	**1**	**5**	**8**	**11**	**15**	**30**	**45**
ROC-AUC	**0,9047**	**0,9509**	**0,966**	**0,9698**	**0,9698**	**0,9792**	**0,9774**
Sensibiliad	0,96226	0,98113	0,98113	0,98113	0,98113	0,98113	0,98113
Especificidad	0,20000	0,60000	0,60000	0,60000	0,60000	0,60000	0,60000
Kappa	0,16226	0,58113	0,58113	0,58113	0,58113	0,58113	0,58113
Precisión	0,92793	0,96396	0,96396	0,96396	0,96396	0,96396	0,96396
Sensib.-macro	0,58113	0,79057	0,79057	0,79057	0,79057	0,79057	0,79057
Exactitud	0,92793	0,96396	0,96396	0,96396	0,96396	0,96396	0,96396
F1	0,92793	0,96396	0,96396	0,96396	0,96396	0,96396	0,96396

Tabla 5.4 Valor *n_neighbors* algoritmo KNN conjunto de datos Lympho.

OPTDIGITS								
n_neighbors	**1**	**5**	**20**	**50**	**80**	**100**	**300**	**1000**
ROC-AUC	**0,3978**	**0,4206**	**0,4115**	**0,467**	**0,5894**	**0,693**	**0.6753**	**0.5697**
Sensibiliad	0,97161	0,97161	0,97161	0,97161	0,97161	0,97161	0,97161	0,97161
Especificidad	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000
Kappa	-0,02839	-0,02839	-0,02839	-0,02839	-0,02839	-0,02839	-0,02839	-0,02839
Precisión	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479
Sensib.-macro	0,48580	0,48580	0,48580	0,48580	0,48580	0,48580	0,48580	0,48580
Exactitud	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479
F1	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479

Tabla 5.5 Valor *n_neighbors* algoritmo KNN conjunto de datos Optdigits.

MAMMOGRAPHY								
n_neighbors	**1**	**5**	**10**	**20**	**40**	**80**	**150**	**500**
ROC-AUC	**0.8047**	**0.839**	**0.8461**	**0.8483**	**0.8495**	**0.8485**	**0.8467**	**0.8479**
Sensibiliad	0,98046	0,98132	0,98156	0,98193	0,98242	0,98230	0,98254	0,98193
Especificidad	0,18782	0,22335	0,23350	0,24873	0,26904	0,26396	0,27411	0,24873
Kappa	0,16828	0,20467	0,21507	0,23066	0,25145	0,24625	0,25665	0,23066
Precisión	0,96185	0,96351	0,96399	0,96471	0,96566	0,96542	0,96590	0,96471
Sensib.-macro	0,58414	0,60233	0,60753	0,61533	0,62573	0,62313	0,62833	0,61533
Exactitud	0,96185	0,96351	0,96399	0,96471	0,96566	0,96542	0,96590	0,96471
F1	0,96185	0,96351	0,96399	0,96471	0,96566	0,96542	0,96590	0,96471

Tabla 5.6 Valor *n_neighbors* algoritmo KNN conjunto de datos Mammography

5.3.2 LOF

LYMPHO							
n_neighbors	**1**	**5**	**8**	**11**	**15**	**30**	**45**
ROC-AUC	**0,4972**	**0,8547**	**0,9358**	**0,9585**	**0,966**	**0,9736**	**0,9755**
Sensibiliad	0,95283	0,97170	0,98113	0,98113	0,98113	0,98113	0,98113
Especificidad	0,00000	0,40000	0,60000	0,60000	0,60000	0,60000	0,60000
Kappa	-0,04717	0,37170	0,58113	0,58113	0,58113	0,58113	0,58113
Precisión	0,90991	0,94595	0,96396	0,96396	0,96396	0,96396	0,96396
Sensib.-macro	0,47642	0,68585	0,79057	0,79057	0,79057	0,79057	0,79057
Exactitud	0,90991	0,94595	0,96396	0,96396	0,96396	0,96396	0,96396
F1	0,90991	0,94595	0,96396	0,96396	0,96396	0,96396	0,96396

Tabla 5.7 Valor *n_neighbors* algoritmo LOF conjunto de datos Lympho.

OPDIGITS								
n_neighbors	**1**	**5**	**20**	**50**	**80**	**100**	**300**	**1000**
ROC-AUC	**0,4931**	**0,5221**	**0,4688**	**0,3691**	**0,3409**	**0,4535**	**0,7615**	**0,5856**
Sensibiliad	0,97345	0,97292	0,97266	0,97161	0,97161	0,97161	0,97161	0,97161
Especificidad	0,06481	0,04630	0,03704	0,00000	0,00000	0,00000	0,00000	0,00000
Kappa	0,03826	0,01922	0,00970	-0,02839	-0,02839	-0,02839	-0,02839	-0,02839
Precisión	0,94836	0,94734	0,94683	0,94479	0,94479	0,94479	0,94479	0,94479
Sensib.-macro	0,51913	0,50961	0,50485	0,48580	0,48580	0,48580	0,48580	0,48580
Exactitud	0,94836	0,94734	0,94683	0,94479	0,94479	0,94479	0,94479	0,94479
F1	0,94836	0,94734	0,94683	0,94479	0,94479	0,94479	0,94479	0,94479

Tabla 5.8 Valor *n_neighbors* algoritmo LOF conjunto de datos Optdigits.

n_neighbors	**3**
ROC-AUC	**0,5333**
Sensibiliad	0,97345
Especificidad	0,06481
Kappa	0,03826
Precisión	0,94836
Sensib.-macro	0,51913
Exactitud	0,94836
F1	0,94836

Tabla 5.9 Valor *n_neighbors* segunda ejecución algoritmo LOF conjunto de datos Optdigits.

MAMMOGRAPHY								
n_neighbors	**1**	**5**	**10**	**20**	**40**	**80**	**150**	**500**
ROC-AUC	**0,5987**	**0,6627**	**0,6899**	**0,7238**	**0,7672**	**0,8088**	**0,8293**	**0,8293**
Sensibiliad	0,97790	0,97888	0,97998	0,98095	0,98095	0,98107	0,98193	0,98193
Especificidad	0,08122	0,12183	0,16751	0,20812	0,20812	0,21320	0,24873	0,24873
Kappa	0,05912	0,10070	0,14749	0,18907	0,18907	0,19427	0,23066	0,23066
Precisión	0,95684	0,95875	0,96089	0,96280	0,96280	0,96304	0,96471	0,96471
Sensib.-macro	0,52956	0,55035	0,57374	0,59454	0,59454	0,59714	0,61533	0,61533
Exactitud	0,95684	0,95875	0,96089	0,96280	0,96280	0,96304	0,96471	0,96471
F1	0,95684	0,95875	0,96089	0,96280	0,96280	0,96304	0,96471	0,96471

Tabla 5.10 Valor *n_neighbors* algoritmo LOF conjunto de datos Mammography.

5.3.3 COF

LYMPHO							
n_neighbors	**2**	**5**	**8**	**11**	**15**	**30**	**45**
ROC-AUC	**0,5358**	**0,7887**	**0,8321**	**0,8151**	**0,8585**	**0,9151**	**0,9057**
Sensibiliad	0,95283	0,95283	0,98113	0,98113	0,98113	0,98113	0,98113
Especificidad	0,00000	0,00000	0,60000	0,60000	0,60000	0,60000	0,60000
Kappa	-0,04717	-0,04717	0,58113	0,58113	0,58113	0,58113	0,58113
Precisión	0,90991	0,90991	0,96396	0,96396	0,96396	0,96396	0,96396
Sensib.-macro	0,47642	0,47642	0,79057	0,79057	0,79057	0,79057	0,79057
Exactitud	0,90991	0,90991	0,96396	0,96396	0,96396	0,96396	0,96396
F1	0,90991	0,90991	0,96396	0,96396	0,96396	0,96396	0,96396

Tabla 5.11 Valor *n_neighbors* algoritmo COF conjunto de datos Lympho.

OPDIGITS							
n_neighbors	**2**	**5**	**20**	**50**	**80**	**100**	**300**
ROC-AUC	**0,5055**	**0,4877**	**0,4585**	**0,4063**	**0,3333**	**0,2596**	**0,3587**
Sensibiliad	0,97529	0,97292	0,97187	0,97161	0,97161	0,97161	0,97161
Especificidad	0,12963	0,04630	0,00926	0,00000	0,00000	0,00000	0,00000
Kappa	0,10492	0,01922	-0,01887	-0,02839	-0,02839	-0,02839	-0,02839
Precisión	0,95194	0,94734	0,94530	0,94479	0,94479	0,94479	0,94479
Sensib.-macro	0,55246	0,50961	0,49057	0,48580	0,48580	0,48580	0,48580
Exactitud	0,95194	0,94734	0,94530	0,94479	0,94479	0,94479	0,94479
F1	0,95194	0,94734	0,94530	0,94479	0,94479	0,94479	0,94479

Tabla 5.12 Valor *n_neighbors* algoritmo COF conjunto de datos Optdigits

MAMMOGRAPHY								
n_neighbors	**2**	**5**	**10**	**20**	**40**	**80**	**150**	**500**
ROC-AUC	**0,6662**	**0,6866**	**0,7208**	**0,7144**	**0,7405**	**0,749**	**0,7814**	**0,7731**
Sensibiliad	0,97802	0,97802	0,97900	0,97949	0,98059	0,98107	0,98071	0,97985
Especificidad	0,08629	0,08629	0,12690	0,14721	0,19289	0,21320	0,19797	0,16244
Kappa	0,06432	0,06432	0,10590	0,12670	0,17348	0,19427	0,17868	0,14229
Precisión	0,95708	0,95708	0,95898	0,95994	0,96208	0,96304	0,96232	0,96065
Sensib.-macro	0,53216	0,53216	0,55295	0,56335	0,58674	0,59714	0,58934	0,57115
Exactitud	0,95708	0,95708	0,95898	0,95994	0,96208	0,96304	0,96232	0,96065
F1	0,95708	0,95708	0,95898	0,95994	0,96208	0,96304	0,96232	0,96065

Tabla 5.13 Valor *n_neighbors* algoritmo COF conjunto de datos Mammography.

5.3.4 HBOS

n_bins: El parámetro seleccionado para las pruebas es *n*_bins o como se había comentado teóricamente en 4.2.3.1 el número de contenedores que formará el histograma.

LYMPHO									
n_bins	**1**	**3**	**5**	**8**	**10**	**12**	**15**	**18**	**50**
ROC-AUC	**0,5**	**0,9962**	**0,9943**	**0,9962**	**0,9962**	**1**	**0,9981**	**0,9925**	**0,9962**
Sensibiliad	1	0,99057	0,990566	0,990566	0,990566	1	0,99056604	0,990566	0,990566038
Especificidad	0	0,8	0,8	0,8	0,8	1	0,8	0,8	0,8
Kappa	0	0,79057	0,790566	0,790566	0,790566	1	0,79056604	0,790566	0,790566038
Precisión	0,954955	0,98198	0,981982	0,981982	0,981982	1	0,98198198	0,981982	0,981981982
Sensib.-macro	0,5	0,89528	0,895283	0,895283	0,895283	1	0,89528302	0,895283	0,895283019
Exactitud	0,911939	0,98198	0,981982	0,981982	0,981982	1	0,98198198	0,981982	0,981981982
F1	0,932951	0,98198	0,981982	0,981982	0,981982	1	0,98198198	0,981982	0,981981982

Tabla 5.14 Valor *n_bins* algoritmo HBOS conjunto de datos Lympho.

OPTDIGITS											
n_bins	**2**	**5**	**10**	**15**	**30**	**60**	**120**	**240**	**480**	**1000**	**2000**
ROC-AUC	**0,5923**	**0,8029**	**0,8571**	**0,8619**	**0,875**	**0,8257**	**0,7951**	**0,7583**	**0,738**	**0,7179**	**0,7051**
Sensibiliad	0,97292	0,97739	0,97792	0,97871	0,97660	0,97503	0,97476	0,97424	0,97371	0,97424	0,97424
Especificidad	0,04630	0,20370	0,22222	0,25000	0,17593	0,12037	0,11111	0,09259	0,07407	0,09259	0,09259
Kappa	0,01922	0,18110	0,20014	0,22871	0,15253	0,09540	0,08587	0,06683	0,04779	0,06683	0,06683
Precisión	0,94734	0,95603	0,95706	0,95859	0,95450	0,95143	0,95092	0,94990	0,94888	0,94990	0,94990
Sensib.-macro	0,50961	0,59055	0,60007	0,61435	0,57626	0,54770	0,54294	0,53342	0,52389	0,53342	0,53342
Exactitud	0,94734	0,95603	0,95706	0,95859	0,95450	0,95143	0,95092	0,94990	0,94888	0,94990	0,94990
F1	0,94734	0,95603	0,95706	0,95859	0,95450	0,95143	0,95092	0,94990	0,94888	0,94990	0,94990

Tabla 5.15 Valor *n_bins* algoritmo HBOS conjunto de datos Optdigits.

MAMMOGRAPHY											
n_bins	2	5	10	15	30	60	120	240	480	1000	2000
ROC-AUC	0,708	0,8145	0,833	0,8178	0,8093	0,7936	0,7959	0,8107	0,821	0,8287	0,8235
Sensibiliad	0,981441	0,98046	0,978999	0,9796093	0,9797314	0,9787546	0,97777778	0,977534	0,979120879	0,97998	0,979365
Especificidad	0,015228	0,17259	0,126904	0,1573604	0,1624365	0,1218274	0,08121827	0,071066	0,137055838	0,17259	0,147208
Kappa	-0,00372	0,1542	0,105902	0,1366312	0,1418166	0,1003335	0,05885028	0,048479	0,115889656	0,15219	0,12626
Precisión	0,958746	0,96149	0,958984	0,9602957	0,9605342	0,9586264	0,95671873	0,956242	0,959341839	0,96101	0,959819
Sensib.-macro	0,498335	0,57653	0,552951	0,5684848	0,571084	0,550291	0,52949803	0,5243	0,558088358	0,57628	0,563287
Exactitud	0,953953	0,9612	0,958984	0,9603938	0,9606317	0,9587287	0,95682569	0,95635	0,959442337	0,96111	0,959918
F1	0,956337	0,96134	0,958984	0,9603447	0,9605829	0,9586776	0,9567722	0,956296	0,959392063	0,96106	0,959868

Tabla 5.16 Valor *n_bins* algoritmo HBOS conjunto de datos Mammography.

n_bins	35
ROC-AUC	0,8056
Sensibiliad	0,979976
Especificidad	0,172589
Kappa	0,152187
Precisión	0,961011
Sensib.-macro	0,576282
Exactitud	0,961107
F1	0,961059

Tabla 5.17 Valor *n_bins* segunda ejecucuón algoritmo HBOS conjunto de datos Mammography.

5.3.5 CBLOF

El parámetro seleccionado para las pruebas es *n_clusters*, que indica el número de grupos que se formarán para detectar las áreas densas y seleccionar los datos que queden fuera de ellas marcándolos como anomalías.

LYMPHO							
n_clusters	4	6	8	10	40	50	60
ROC-AUC	0,9755	0,9698	0,9585	0,9432	0,5613	0,6821	0,6736
Sensibiliad	0,98113	0,98113	0,98113	0,98113	0,95283	0,96226	0,97170
Especificidad	0,60000	0,60000	0,60000	0,60000	0,00000	0,20000	0,40000
Kappa	0,58113	0,58113	0,58113	0,58113	-0,04717	0,16226	0,37170
Precisión	0,96396	0,96396	0,96396	0,96396	0,90991	0,92793	0,94595
Sensib.-macro	0,79057	0,79057	0,79057	0,79057	0,47642	0,58113	0,68585
Exactitud	0,96396	0,96396	0,96396	0,96396	0,90991	0,92793	0,94595
F1	0,96396	0,96396	0,96396	0,96396	0,90991	0,92793	0,94595

Tabla 5.18 Valor *n_clusters* algoritmo CBLOF conjunto de datos Lympho.

OPTDIGITS				
n_clusters	**7**	**8**	**10**	**30**
ROC-AUC	**0,787**	**0,8073**	**0,8066**	**0,2661**
Sensibiliad	0,97161	0,97161	0,97161	0,97161
Especificidad	0,00000	0,00000	0,00000	0,00000
Kappa	-0,02839	-0,02839	-0,02839	-0,02839
Precisión	0,94479	0,94479	0,94479	0,94479
Sensib.-macro	0,48580	0,48580	0,48580	0,48580
Exactitud	0,94479	0,94479	0,94479	0,94479
F1	0,94479	0,94479	0,94479	0,94479

Tabla 5.19 Valor *n_clusters* algoritmo CBLOF conjunto de datos Optdigits.

MAMMOGRAPHY										
n_clusters	**4**	**6**	**8**	**10**	**12**	**20**	**50**	**150**	**500**	**1000**
ROC-AUC	**0,8803**	**0,755**	**0,7885**	**0,8109**	**0,8295**	**0,8307**	**0,8358**	**0,84510**	**0,8131**	**0,7912**
Sensibiliad	0,98376	0,97814	0,98230	0,97998	0,98071	0,98156	0,98205	0,98339	0,98144	0,98022
Especificidad	0,32487	0,09137	0,26396	0,16751	0,19797	0,23350	0,25381	0,30964	0,22843	0,17766
Kappa	0,30863	0,06951	0,24625	0,14749	0,17868	0,21507	0,23586	0,29304	0,20987	0,15788
Precisión	0,96828	0,95731	0,96542	0,96089	0,96232	0,96399	0,96495	0,96757	0,96375	0,96137
Sensib.-macro	0,65432	0,53476	0,62313	0,57374	0,58934	0,60753	0,61793	0,64652	0,60493	0,57894
Exactitud	0,96828	0,95731	0,96542	0,96089	0,96232	0,96399	0,96495	0,96757	0,96375	0,96137
F1	0,96828	0,95731	0,96542	0,96089	0,96232	0,96399	0,96495	0,96757	0,96375	0,96137

Tabla 5.20 Valor *n_clusters* algoritmo CBLOF conjunto de datos Mammography.

5.3.6 PCA

El parámetro seleccionado para las pruebas es *n_components*, que indica el número de componentes principales a los que se reducirá la dimensión de las muestras del conjunto de datos, y se diferenciará entre punto anómalo o normal en función de si forma parte del componente principal o no..

LYMPHO						
n_components	**1**	**3**	**5**	**8**	**10**	**14**
ROC-AUC	**0,9943**	**0,9868**	**0,9849**	**0,983**	**0,9811**	**0,9811**
Sensibiliad	1,00000	0,99057	0,99057	0,99057	0,99057	0,99057
Especificidad	0,80000	0,60000	0,60000	0,60000	0,60000	0,60000
Kappa	0,88425	0,65276	0,65276	0,65276	0,65276	0,65276
Precisión	0,99099	0,97297	0,97297	0,97297	0,97297	0,97297
Sensib.-macro	0,90000	0,79528	0,79528	0,79528	0,79528	0,79528
Exactitud	0,99108	0,97089	0,97089	0,97089	0,97089	0,97089
F1	0,99051	0,97153	0,97153	0,97153	0,97153	0,97153

Tabla 5.21 Valor n_components algoritmo PCA conjunto de datos Lympho.

OPTDIGITS							
n_components	**1**	**5**	**10**	**15**	**20**	**25**	**40**
ROC-AUC	**0,4856**	**0,4774**	**0,5026**	**0,5362**	**0,5426**	**0,5348**	**0,5377**
Sensibiliad	0,97161	0,97161	0,97161	0,97161	0,97161	0,97161	0,97161
Especificidad	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000
Kappa	-0,02839	-0,02839	-0,02839	-0,02839	-0,02839	-0,02839	-0,02839
Precisión	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479
Sensib.-macro	0,48580	0,48580	0,48580	0,48580	0,48580	0,48580	0,48580
Exactitud	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479
F1	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479	0,94479

Tabla 5.22 Valor *n_components* algoritmo PCA conjunto de datos Optdigits.

MAMMOGRAPHY						
n_components	**1**	**2**	**3**	**4**	**5**	**6**
ROC-AUC	**0,7304**	**0,8479**	**0,8805**	**0,8832**	**0,8746**	**0,883**
Sensibiliad	0,97790	0,97961	0,97961	0,97863	0,97839	0,97814
Especificidad	0,24873	0,31980	0,31980	0,27919	0,26904	0,25888
Kappa	0,20951	0,27678	0,27678	0,23834	0,22873	0,21912
Precisión	0,96077	0,96411	0,96411	0,96220	0,96173	0,96125
Sensib.-macro	0,61332	0,64970	0,64970	0,62891	0,62371	0,61851
Exactitud	0,96380	0,96690	0,96690	0,96513	0,96468	0,96424
F1	0,96225	0,96546	0,96546	0,96363	0,96317	0,96271

Tabla 5.23 Valor *n_components* algoritmo PCA conjunto de datos Mammography.

5.3.7 IForest

El parámetro seleccionado para las pruebas es *n_estimators*, que indica el número de nodos base que tendrá el árbol creado a través del conjunto de datos.

LYMPHO								
n_estimators	**2**	**5**	**8**	**10**	**12**	**15**	**30**	**50**
ROC-AUC	**0,9783**	**0,9283**	**0,9123**	**0,9811**	**0,9962**	**1**	**0,9755**	**0,9943**
Sensibiliad	0,99057	0,98113	0,98113	0,98113	0,99057	1,00000	0,98113	0,99057
Especificidad	0,60000	0,60000	0,60000	0,60000	0,80000	1,00000	0,60000	0,80000
Kappa	0,65276	0,58113	0,58113	0,58113	0,79057	1,00000	0,58113	0,79057
Precisión	0,97297	0,96396	0,96396	0,96396	0,98198	1,00000	0,96396	0,98198
Sensib.-macro	0,79528	0,79057	0,79057	0,79057	0,89528	1,00000	0,79057	0,89528
Exactitud	0,97089	0,96396	0,96396	0,96396	0,98198	1,00000	0,96396	0,98198
F1	0,97153	0,96396	0,96396	0,96396	0,98198	1,00000	0,96396	0,98198

Tabla 5.24 Valor *n_estimators* algoritmo IForest conjunto de datos Lympho.

OPTDIGITS								
n_estimators	**2**	**5**	**8**	**10**	**30**	**50**	**80**	**100**
ROC-AUC	**0,5082**	**0,8055**	**0,6892**	**0,731**	**0,635**	**0,8234**	**0,7532**	**0,7513**
Sensibiliad	0,97950	0,97345	0,97187	0,97187	0,97187	0,97424	0,97266	0,97266
Especificidad	0,03704	0,05556	0,00926	0,00926	0,00926	0,09259	0,03704	0,03704
Kappa	0,01872	0,02914	-0,01887	-0,01887	-0,01887	0,06683	0,00970	0,00970
Precisión	0,95348	0,94811	0,94530	0,94530	0,94530	0,94990	0,94683	0,94683
Sensib.-macro	0,50827	0,51450	0,49057	0,49057	0,49057	0,53342	0,50485	0,50485
Exactitud	0,94733	0,94787	0,94530	0,94530	0,94530	0,94990	0,94683	0,94683
F1	0,95037	0,94799	0,94530	0,94530	0,94530	0,94990	0,94683	0,94683

Tabla 5.25 Valor *n_estimators* algoritmo IForest conjunto de datos Optdigits.

MAMMOGRAPHY								
n_estimators	**5**	**10**	**50**	**100**	**200**	**500**	**1000**	**3000**
ROC-AUC	**0,8631**	**0,8721**	**0,8619**	**0,8421**	**0,8674**	**0,8576**	**0,861**	**0,8603**
Sensibiliad	0,98388	0,98388	0,98181	0,98156	0,98193	0,98168	0,98168	0,98168
Especificidad	0,20812	0,20812	0,24365	0,23350	0,24873	0,23858	0,23858	0,23858
Kappa	0,20414	0,20414	0,22546	0,21507	0,23066	0,22026	0,22026	0,22026
Precisión	0,96566	0,96566	0,96447	0,96399	0,96471	0,96423	0,96423	0,96423
Sensib.-macro	0,59600	0,59600	0,61273	0,60753	0,61533	0,61013	0,61013	0,61013
Exactitud	0,96353	0,96353	0,96447	0,96399	0,96471	0,96423	0,96423	0,96423
F1	0,96457	0,96457	0,96447	0,96399	0,96471	0,96423	0,96423	0,96423

Tabla 5.26 Valor *n_estimators* algoritmo IForest conjunto de datos Mammography.

5.4. Ejecución algoritmos sobre el conjunto de prueba

Ejecución de los algoritmos sobre el conjunto de prueba con parámetros adaptados para un mejor rendimiento. Reflejado en el flujo de trabajo en la Fase 3. En esta fase se ejecuta el algoritmo entrenado mediante el entrenamiento para el conjunto de prueba obteniendo como resultados las mismas métricas mostradas en el conjunto de entrenamiento. La ejecución se realiza mediante los valores de los parámetros obtenidos en la fase 3 y a continuación se muestran los resultados para cada algoritmo y conjunto de datos.

	LYMPHO						
	KNN	LOF	COF	CBLOF	Iforest	HBOS	PCA
ROC-AUC	1	1	1	1	1	1	1
Sensibiliad	1	1	1	1	1	1	1
Especificidad	1	1	0	1	1	1	1
Kappa	1	1	0	1	1	1	1
Precisión	1	1	0,9730	1	1	1	1
Sensib.-macro	1	1	0,5	1	1	1	1
Exactitud	1	1	0,9467	1	1	1	1
F1	1	1	0,9596	1	1	1	1

Tabla 5.27 Resultados métricas de todos los algoritmos para conjunto de datos Lympho.

	OPDIGITS						
	KNN	LOF	COF	CBLOF	Iforest	HBOS	PCA
ROC-AUC	0,6186	0,4689	0,7116	0,7121	0,6200	0,8976	0,4723
Sensibiliad	0,9746	0,9770	0,9620	0,9707	0,9612	0,9770	0,9739
Especificidad	0,0000	0,0238	0,4286	0,0000	0,0238	0,1905	0,0000
Kappa	-0,0287	0,0010	0,3060	-0,0311	-0,0138	0,1777	-0,0292
Precisión	0,9433	0,9463	0,9448	0,9394	0,9310	0,9517	0,9425
Sensib.-macro	0,4873	0,5004	0,6953	0,4853	0,4925	0,5837	0,4869
Exactitud	0,9358	0,9377	0,9578	0,9357	0,9368	0,9488	0,9358
F1	0,9395	0,9420	0,9507	0,9376	0,9339	0,9502	0,9391

Tabla 5.28 Resultados métricas de todos los algoritmos para conjunto de datos Optdigits.

	MAMMOGRAPHY						
	KNN	LOF	COF	CBLOF	Iforest	HBOS	PCA
ROC-AUC	0,8606	0,8355	0,7748	0,8312	0,8451	0,8246	0,9017
Sensibiliad	0,9795	0,9769	0,9579	0,9795	0,9737	0,9715	0,9791
Especificidad	0,3492	0,3175	0,3016	0,3016	0,1905	0,1587	0,3492
Kappa	0,2945	0,2549	0,1674	0,2572	0,1411	0,1091	0,2921
Precisión	0,9653	0,9621	0,9431	0,9642	0,9560	0,9531	0,9649
Sensib.-macro	0,6644	0,6472	0,6298	0,6405	0,5821	0,5651	0,6642
Exactitud	0,9691	0,9674	0,9645	0,9674	0,9623	0,9609	0,9690
F1	0,9671	0,9646	0,9530	0,9658	0,9591	0,9569	0,9669

Tabla 5.29 Resultados métricas de todos los algoritmos para conjunto de datos Mammography.

A partir de estos resultados se realiza el siguiente capítulo de análisis.

CAPITULO 6: Análisis

Se evalúa en primera instancia los resultados a partir de las métricas seleccionadas. Se tiene en cuenta para la comparación la salida de la ejecución de los algoritmos para el subconjunto de prueba en cada conjunto de datos.

6.1 Análisis conjunto de datos Lympho

Con la excepción del algoritmo COF, el resto de los algoritmos son capaces de encontrar y detectar la única anomalía del conjunto de datos. Esto supone por lo tanto para todos estos algoritmos un valor = 1 en todas las métricas evaluadas.

En el caso de COF, revisando tanto los resultados como la matriz de confusión devuelta:

VP = 36 FN = 0 FP = 1 VN = 0

Se observa que no es capaz de identificar el valor anómalo contenido en el conjunto de datos y por lo tanto su especificidad será 0.

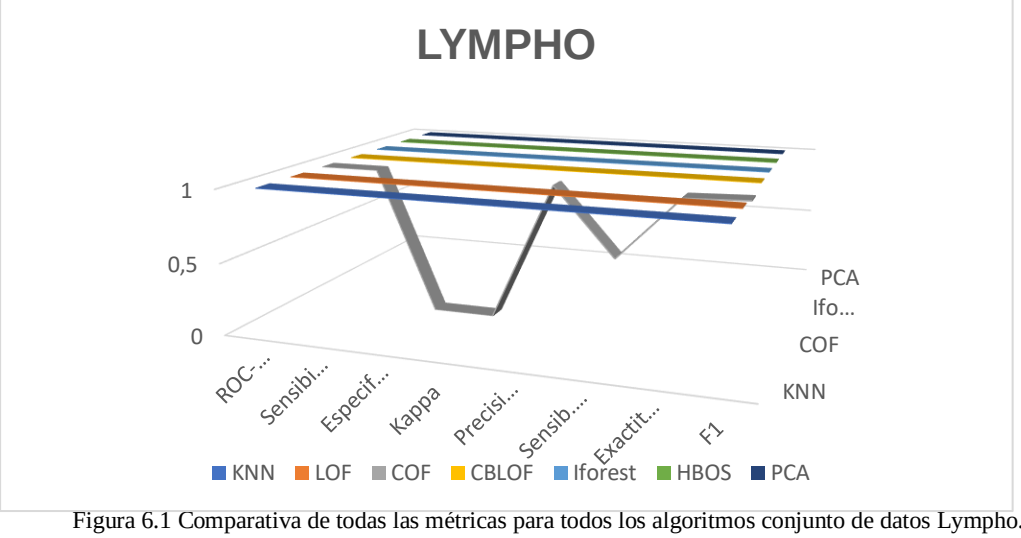

Figura 6.1 Comparativa de todas las métricas para todos los algoritmos conjunto de datos Lympho.

Se concluye que para un conjunto de datos con un tamaño pequeño de muestras tras un preparado de los datos y un buen ajuste de parámetros los algoritmos tienen un rendimiento perfecto pese al pequeño porcentaje de datos anómalo.

6.2 Análisis conjunto de datos Optdigits

Todos los algoritmos obtienen una sensibilidad superior a 0.96. Esto indica que para el conjunto de datos todos los algoritmos tienen una proporción de casos positivos correctos muy alta. Este dato no es muy significativo para la evaluación del algoritmo debido al gran número de muestras y el pequeño porcentaje de anomalías. Ocurre lo mismo para el valor de la exactitud, dado que representa una buena puntuación (ningún algoritmo por debajo de 0,93) debido al gran número de verdaderos positivos detectados. Con una especificidad muy baja para la mayoría de los algoritmos, COF es el que mejor rendimiento da en este aspecto. Es capaz de reconocer 18 anomalías correctamente dando una especificidad de 0,4286. HBOS es el segundo algoritmo con mejor especificidad con un valor de 0,1905.

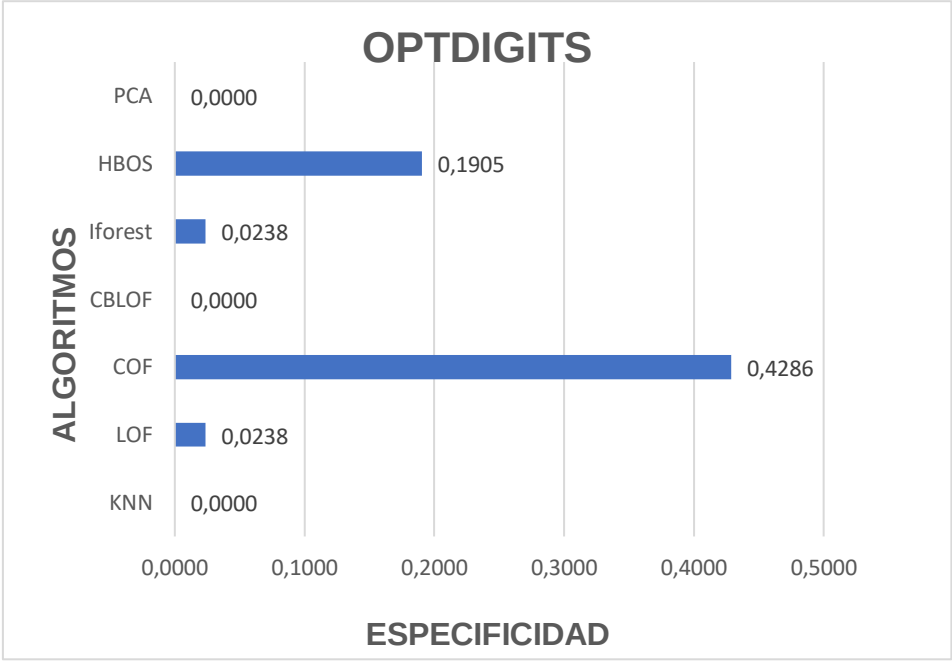

Figura 6.2 Valores especificidad todos los algoritmos conjuntos de datos Optdigits.

Mediante los valores de kappa obtenidos nos reafirmamos en que ningún algoritmo tiene un buen acuerdo entre los clasificadores, de hecho, su acuerdo es totalmente aleatorio a excepción de nuevo de HBOS con un resultado de 0,1778 y COF con un resultado de 0,3060. En cuanto a la medida ROC-AUC, el algoritmo PCA tiene un muy buen rendimiento con un valor de 0,9017 que indica su gran capacidad de separabilidad entre clases. Este algoritmo es muy capaz de distinguir entre la clase positiva y la clase negativa. Por el contrario, COF tiene el menor valor de ROC-AUC indicando una dudosa capacidad de separar entre clases.

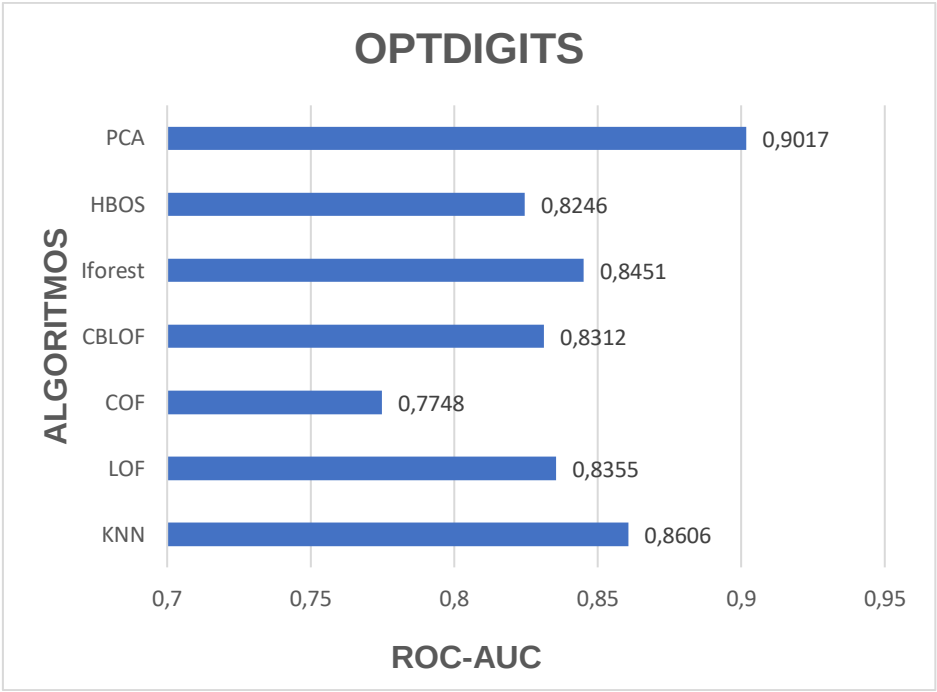

Figura 6.3 Valores ROC-AUC todos los algoritmos conjuntos de datos Optdigits.

Observando la medida de sensibilidad macro, confirmamos que el algoritmo COF con un valor de 0,6953 es el más capacitado para detectar los casos positivos independientemente del desequilibrio de las clases. De nuevo junto a HBOS con un valor de 0,5837 ambos algoritmos destacan en su rendimiento frente al resto.

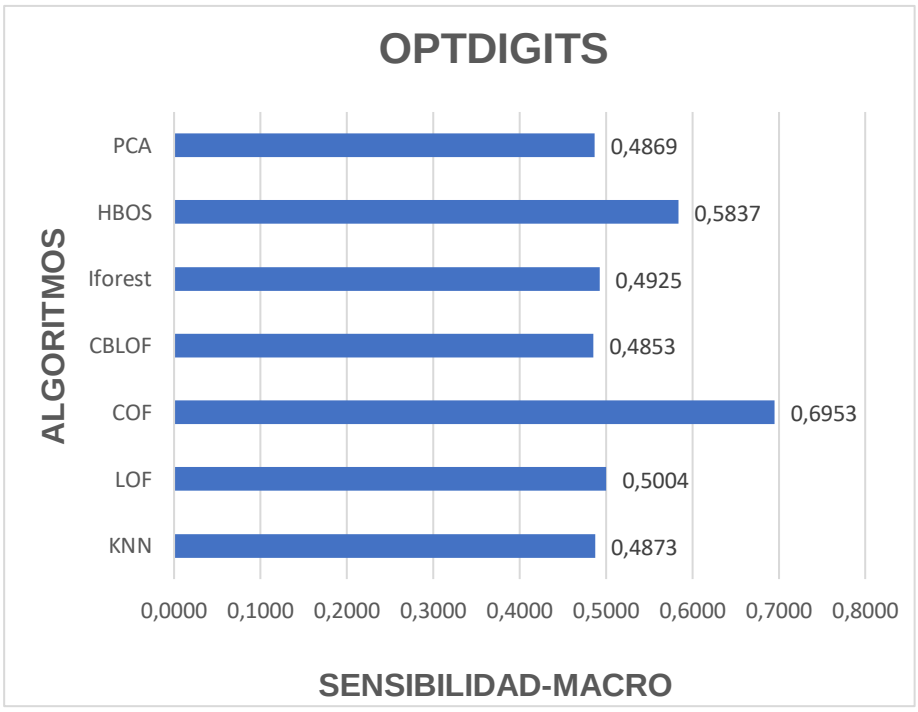

Figura 6.4 Valores sensibilidad-macro todos los algoritmos conjuntos de datos Optdigits.

El buen rendimiento del algoritmo COF para la detección de anomalías conlleva a plantear la comparación a la ejecución del código sin el previo ajuste de parámetros tomando los valores por defecto tanto en el parámetro *contamination* como en el parámetro *n_neighbors*.

En la figura 6.5 se muestra una comparación de resultados antes del ajuste y después del ajuste de parámetros.

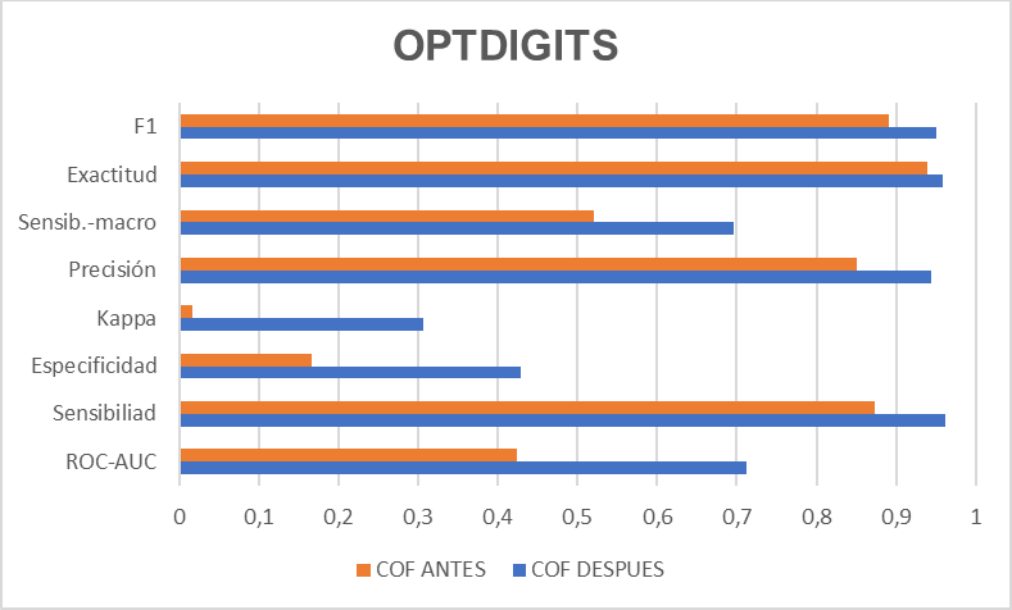

Figura 6.5 Comparativa de los resultados conjunto de datos Optdigits antes y después de ajuste de parámetros.

Se muestra así la alta mejora de rendimiento mediante el ajuste de parámetros suponiendo un aumento de más del 50% en la especificidad y más de un 75% en la métrica kappa.

Se concluye, por lo tanto, que para este conjunto de datos el mejor algoritmo es COF, seguido de HBOS y que, pese al rendimiento óptimo del conjunto de algoritmos, tanto COF como LOF son los más capacitados para la detección de anomalías en este conjunto de datos. Se muestra que KNN, CBLOF y PCA tienen una sensibilidad de 0 indicando que no son capaces de reconocer a las anomalías. Se demuestra la mejora del rendimiento en los algoritmos mediante el ajuste de parámetros.

6.3 Análisis conjunto de datos Mammography

De nuevo en este conjunto de datos como en Optdigits hay una muy buena sensibilidad. Ningún algoritmo se encuentra por debajo de 0,97. Como en el conjunto de datos anterior el hecho de tener una proporción de casos positivos correctos muy alta no es realmente significativo para la evaluación del algoritmo debido al gran número de muestras y el pequeño porcentaje de anomalías. Mediante la comparación de la métrica Kappa observamos que para el conjunto de algoritmos hay un acuerdo aleatorio entre los clasificadores, especialmente para Iforest con un valor de 0,1411 y HBOS con un valor de 0,1091.

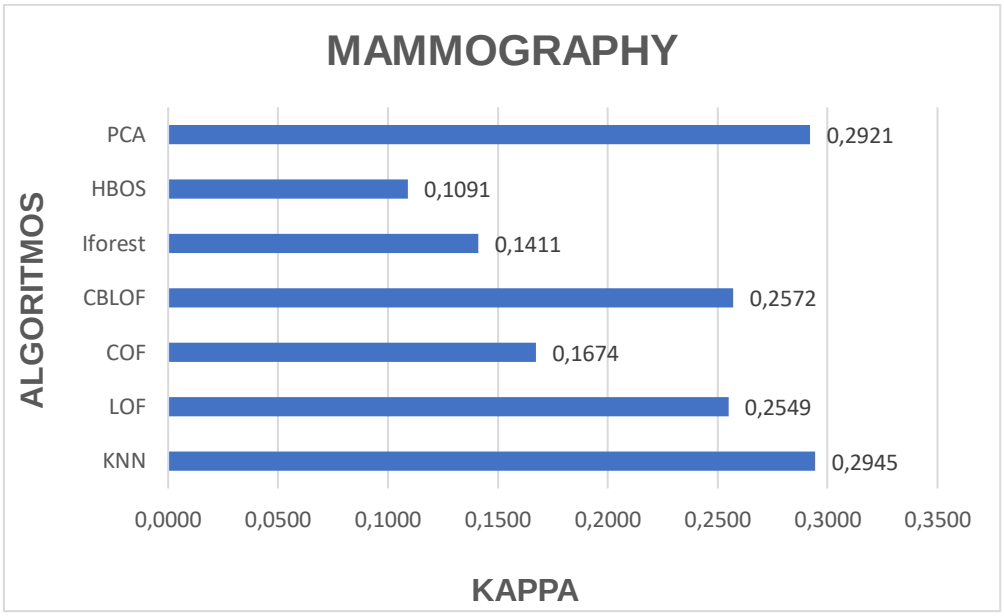

Figura 6.6 Valores kappa todos los algoritmos conjuntos de datos Mammography.

En cuanto a la especificidad de nuevo hablamos de HBOS como el algoritmo con menos capacidad para clasificar los casos negativos correctamente con una puntuación de 0,1587 seguido de Iforest con 0, 1905. Pese a que ningún algoritmo supera 0,3492 como es el caso de PCA y KNN los valores de Iforest y HBOS se encuentran muy por debajo del resto.

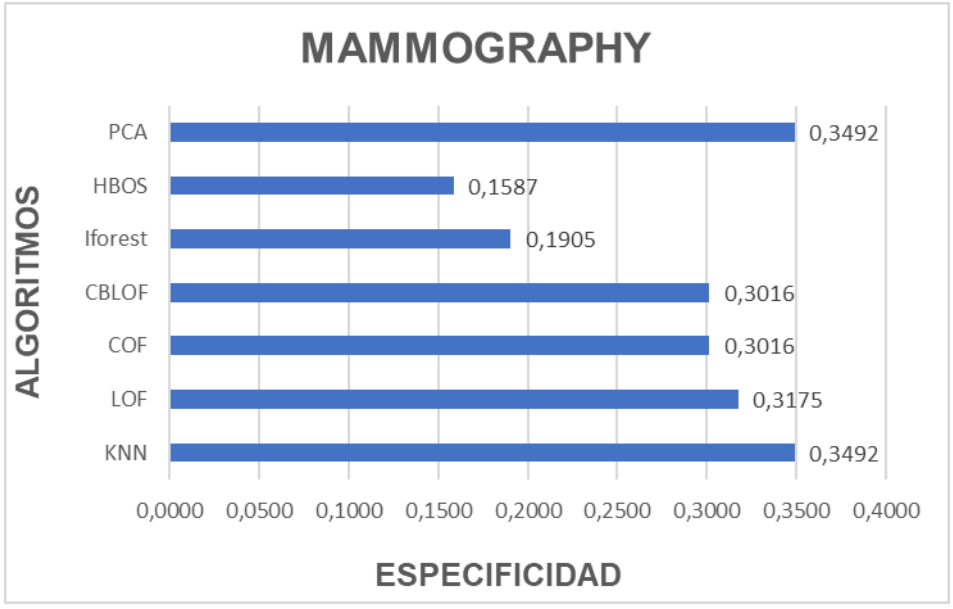

Figura 6.7 Valores especificidad todos los algoritmos conjuntos de datos Mammography.

Se mencionan brevemente las medidas de precisión, exactitud y F1 que demuestran que todos los algoritmos tienen un rendimiento muy bueno, siendo la menor precisión 0,9431 y el menor F1 0,9530 en el caso de COF, y la menor exactitud la obtiene el algoritmo HBOS: 0,9609.

En cuanto a la medida ROC-AUC, el algoritmo PCA tiene un muy buen rendimiento con un valor de 0,9017 que indica su gran capacidad de separabilidad entre clases. Este algoritmo es muy capaz de distinguir entre la clase positiva y la clase negativa. Por el contrario, COF tiene el menor valor de ROC-AUC indicando una dudosa capacidad de separar entre clases.

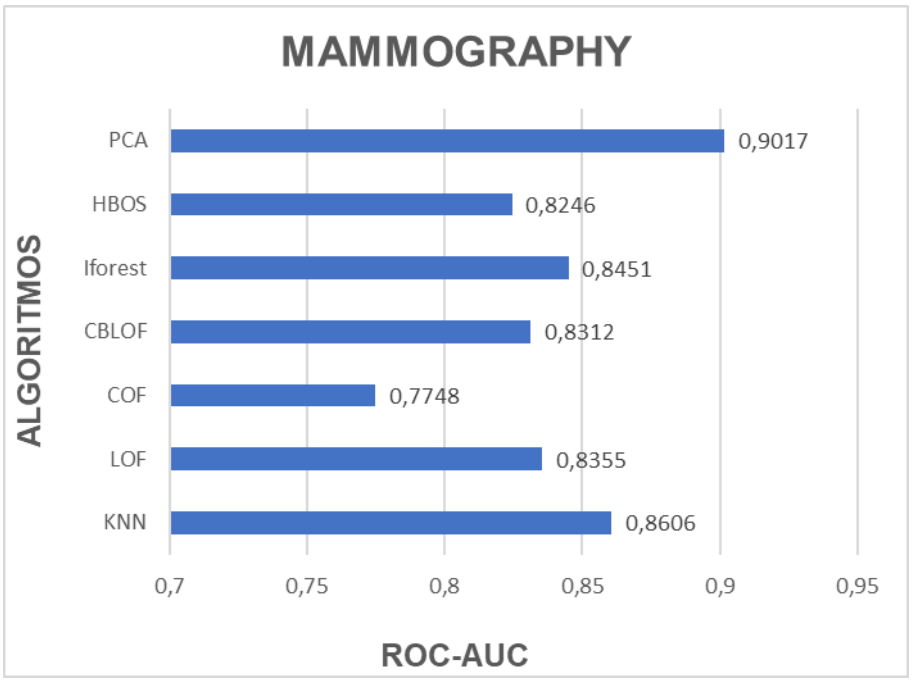

Figura 6.8 Valores ROC-AUC todos los algoritmos conjuntos de datos Mammography.

Se concluye indicando que los algoritmos con un peor rendimiento significativo son Iforest y HBOS, al igual que COF que presenta un rendimiento muy inferior al conjunto de datos comentado anteriormente. El mejor rendimiento es presentado por KNN y PCA presentando la mayor capacidad para la detección de anomalías.

6.4 Análisis global

Se muestra que para un conjunto de datos con un pequeño tamaño de muestras los algoritmos están perfectamente capacitados para la detección de anomalías. Independientemente del conjunto de datos, se observa que para un gran número de muestras el valor de la especificidad es bajo por lo que se demuestra que los algoritmos no son muy capaces de capturar las anomalías. Se valora la misma idea mediante la métrica kappa dado que para los conjuntos de datos con un gran número de datos en ningún algoritmo los evaluadores llegan a estar completamente de acuerdo.

No resulta relevante en cuanto a rendimiento el número de dimensiones. No hay ningún indicio en estos conjuntos de datos si bien el conjunto de datos Lympho tiene un número muy superior de dimensiones y tiene un rendimiento casi idéntico para sus algoritmos con mejor rendimiento comparado con Mammography. Si bien es cierto que se encuentra una diferencia de peso en cuanto a qué tipo de algoritmos convienen según el conjunto de datos como se comenta a continuación.

Dada la gran diferencia en el número de dimensiones entre los conjuntos de datos Mammography y Optidigits, ambos contando con un alto número de muestras, y el rendimiento totalmente adverso en cuanto al algoritmo HBOS, se propone la idea de que su rendimiento es variable en función del número de dimensiones del conjunto de datos.

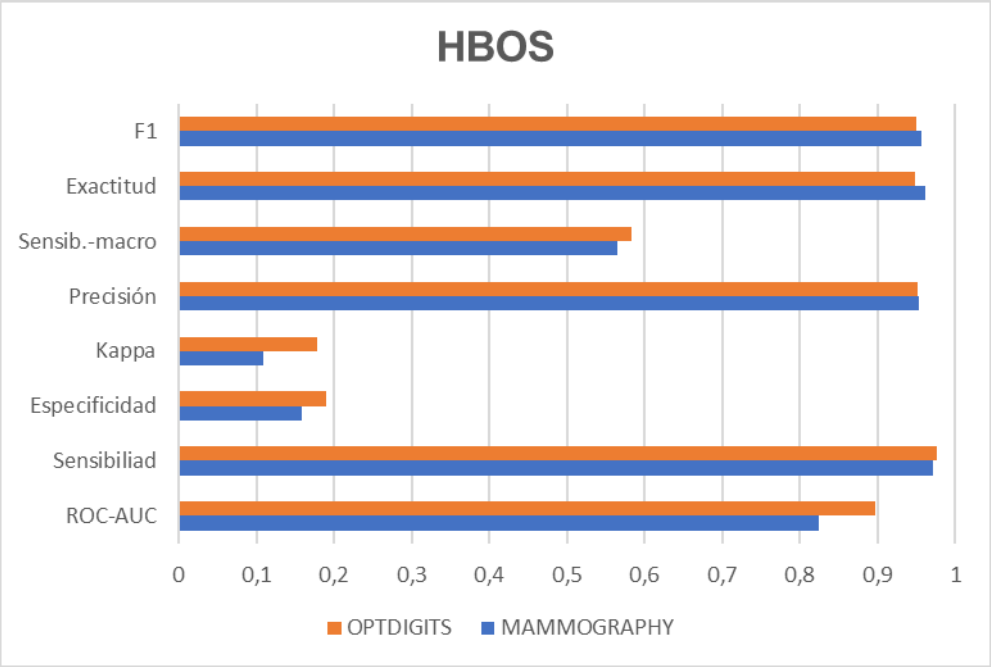

Figura 6.9 Comparativa de algoritmo HBOS para conjunto de datos Optdigits y Mammography.

Dentro de los algoritmos basados en vecinos para un conjunto de datos con un gran número de dimensiones se considera COF como un algoritmo apropiado mientras que para un número reducido de dimensiones tanto KNN como LOF son muy superiores. Se considera PCA como un algoritmo apropiado para un número reducido de dimensiones como el caso de Mammography y muy poco apropiado con un rendimiento totalmente adverso en el caso de un gran número de dimensiones como es el caso de Optdigits. El rendimiento de Iforest no es óptimo para ninguno de los conjuntos de datos con un gran número de muestras.

CAPITULO 7: Conclusiones

En este trabajo se han comparado los principales algoritmos de aprendizaje automático no supervisado en distintos conjuntos de datos. La selección del estudio viene dada por el propósito de evaluar el estado actual de un campo como es la detección de anomalías en distintos dominios. Respecto al análisis de los algoritmos es primordial resaltar el perfecto rendimiento de todos los seleccionados para un conjunto de datos con un pequeño número de muestras. Por otra parte, las observaciones obtenidas sobre los dos conjuntos de datos de mayor tamaño indican que es necesario un buen procesado de los datos, y un buen ajuste de los parámetros a la hora de entrenar los algoritmos. Tras estos pasos la ejecución ha determinado los siguientes hechos:

- Dentro de los algoritmos basados en vecinos la selección del algoritmo debe basarse en el número de dimensiones del conjunto de datos. Se ha comprobado que, para un conjunto de datos con un tamaño reducido en dimensiones, el funcionamiento de los algoritmos KNN y LOF proporcionan un mejor rendimiento. Sin embargo, para un gran número de dimensiones el rendimiento de COF ha sido notablemente superior.

- El algoritmo HBOS es eficiente para un gran número de dimensiones.

- El algoritmo Iforest no ha presentado un buen rendimiento para ninguno de los conjuntos de datos.

- Tanto CBLOF como PCA no han sido capaces de capturar anomalías para un conjunto de datos de altas dimensiones, pero han presentado un rendimiento óptimo con un tamaño pequeño de dimensiones.

En cuanto a las dificultades encontradas en el trabajo se debe mencionar el alto tiempo de cómputo que conllevan los algoritmos. Durante el trabajo se han descartado algoritmos y conjuntos de datos debido al exceso de tiempo de ejecución inaceptable para la evaluación.

Finalmente se debe indicar que las conclusiones son extraídas únicamente de las pruebas realizadas y que pueden alejarse de la realidad para distintos conjuntos de datos.

BIBLIOGRAFÍA

[1] Why AI Anomaly Detection Is Business Critical [Internet]. millimetric.ai. 2020 [visitado 28 mayo 2020]. Disponible en: https://www.millimetric.ai/2020/04/30/why-ai-anomaly-detection-is-business-critical/

[2] Vardon E. How To Apply Anomaly Detection And Reap These Three Benefits [Internet]. forbes. 2020 [visitado 11 abril 2020]. Disponible en: https://www.forbes.com/sites/forbesagencycouncil/2020/02/03/how-to-apply-anomaly-detection-and-reap-these-three-benefits/#5a289a5114bf

[3] Gerbert P, Hecker M, Steinhäuser S, Ruwolt P. Putting Artificial Intelligence to Work [Internet]. bcg. 2017 [visitado 13 abril 2020]. Disponible en: https://www.bcg.com/publications/2017/technology-digital-strategy-putting-artificial-intelligence-work.aspx

[4] https://paperswithcode.com/task/anomaly-detection#code

[5] Real Academia Española. (s.f.). Anomalía. En Diccionario de la lengua española. [visitado 18 de mayo de 2020]. Disponible en:, de https://dle.rae.es/anomal%C3%ADa

[6] J. Gao, F. Liang, W. Fan, C. Wang, Y. Sun, and J. Han. On community outliers and their efficient detection in information networks. In Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 813–822. ACM, 2010.

[7] Kibish S. A note about finding anomalies [Internet]. towardsdatascience. 2018 [visitatdo 17 abril 2020]. Disponible en: https://towardsdatascience.com/a-note-about-finding-anomalies-f9cedee38f0b

[8] Chandola V. Arindam B. Kumar V. Anomaly Detection: A Survey. 2019 [visitado 13 mayo 2020].

[9] Mehrotra, K. G., Mohan, C. K., & Huang, H. (2017). Anomaly detection principles and algorithms (p. 11-18). New York, NY, USA:: Springer International Publishing.

[10] https://dictionary.cambridge.org/es/diccionario/ingles/dataset

[11]Shebuti Rayana (2016). ODDS Library [http://odds.cs.stonybrook.edu]. Stony Brook, NY: Stony Brook University, Department of Computer Science.

[12]] Nautiyal D. Underfitting and Overfitting in Machine Learning [Internet]. GeeksforGeeks. [visitado 17 mayo 2020]. Disponible en: https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/

[13] Bhande A. What is underfitting and Overfitting in Machine Learning and how to deal with it [Internet]. Medium. 2018 [visitado 17 mayo 2020]. Disponible en: https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76

[14] Curso aprendizaje automático intensivo google[visitado 17 mayo 2020]. Disponible en: https://developers.google.com/machine-learning/crash-course/training-and-test-sets/splitting-data

[15] Guinness R. Is there a rule-of-thumb for how to divide a dataset into training and validation sets? [Internet]. stackoverflow. 2013 [visitado 18 mayo 2020]. Disponible en: https://stackoverflow.com/questions/13610074/is-there-a-rule-of-thumb-for-how-to-divide-a-dataset-into-training-and-validatio

[16] Jaitley U. Why Data Normalization is necessary for Machine Learning models [Internet]. Medium. 2018 [visitado 22 mayo 2020]. Disponible en: https://medium.com/@urvashilluniya/why-data-normalization-is-necessary-for-machine-learning-models-681b65a05029#:~:text=Why%20Data%20Normalization%20is%20necessary%20for%20Machine%20Learning%20models,-Urvashi%20Jaitley&text=The%20goal%20of%20normalization%20is,dataset%20does%20not%20require%20normalization.

[17] https://scikit-learn.org/stable/modules/preprocessing.html

[18] Brownlee J. Supervised and Unsupervised Machine Learning Algorithms. Machine Learning Mastery. 2019 [visitado 23 mayo 2020]. Disponible en: https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/

[19] Kibish S. A note about finding anomalies [Internet]. Medium. 2018 [visitado 23 mayo 2020]. Disponible en: https://towardsdatascience.com/a-note-about-finding-anomalies-f9cedee38f0b

[20] A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data
Goldstein M, Uchida S (2016) A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data. PLOS ONE 11(4): e0152173. https://doi.org/10.1371/journal.pone.0152173

[21] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In ACM SIGMOD Record, volume 29, pages 427–438, 2000.

[22]Mennatallah A. Comparison of Unsupervised Anomaly Detection Techniques[Internet].Kaiserslautern,Germany:Media Engineering and Technology
German University in Cairo
and
Multimedia Analysis and Data Mining Competence Center
German Research Center for Artificial Intelligence (DFKI GmbH) ; 2011 [visitado el 8 de junio de 2019]. Disponible en: https://madm.dfki.de/_media/theses/bachelorthesis-amer_2011.pdf

[23] Everitt, Brian S.; Landau, Sabine; Leese, Morven; and Stahl, Daniel (2011) "Miscellaneous Clustering Methods", in Cluster Analysis, 5th Edition, John Wiley & Sons, Ltd., Chichester, UK, 2011.

[24] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and J¨org Sander. Lof: identifying density-based local outliers. In ACM SIGMOD Record, volume 29, pages 93–104, 2000.

[25] Jian Tang, Zhixiang Chen, Ada Wai-Chee Fu, and David W Cheung. Enhancing effectiveness of outlier detections for low density patterns. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 535–548. Springer, 2002.

[26] K.Manoj.Nearest-Neighbor Based Outlier detection in Data Mining,chapter 7,pages 10-14 https://shodhganga.inflibnet.ac.in/bitstream/10603/131060/11/12_chapter7.pdf

[27] Spiros Papadimitriou, Hiroyuki Kitagawa, Phillip B Gibbons, and Christos Faloutsos. Loci: Fast outlier detection using the local correlation integral. In International Conference on Data Engineering (ICDE), pages 315–326. IEEE, 2003.

[28] Amer M, Goldstein M. Nearest-Neighbor and Clustering based Anomaly Detection Algorithms for RapidMiner. In: Simon Fischer IM, editor. Proceedings of the 3rd RapidMiner Community Meeting and Conferernce (RCOMM 2012). Shaker Verlag GmbH; 2012. p. 1–12. 2012.

[29] Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. Pattern Recognition Letters, 24(9-10):1641–1650, 2003.

[30] He Z, Xu X, Deng S. Discovering Cluster-based Local Outliers. Pattern Recognition Letters. 2003;24(9–10):1641–1650.

[31] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In International Conference on Data Mining (ICDM), pages 413–422. IEEE, 2008.

[32] Markus Goldstein and Andreas Dengel. Histogram-based outlier score (HBOS): A fast unsupervised anomaly detection algorithm. Annual German Conference on Artificial Intelligence (KI-2012), pages 59–63, 2012.

[33] Goldstein, Markus & Dengel, Andreas. (2012). Histogram-based Outlier Score (HBOS): A fast Unsupervised Anomaly Detection Algorithm.

[34] Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. A novel anomaly detection scheme based on principal component classifier. Technical report, Department of Electrical and Computer Engineering, University of Miami, 2003.

[35] Fouzi Harrou, Farid Kadri, Sondes Chaabane, Christian Tahon, Ying Sun,
Improved principal component analysis for anomaly detection: Application to an emergency department,Computers & Industrial Engineering,Volume 88,2015,Pages 63-77,ISSN 0360-8352, https://doi.org/10.1016/j.cie.2015.06.020.

[36] J. Edward Jackson & Govind S. Mudholkar (1979) Control Procedures for Residuals Associated With Principal Component Analysis, Technometrics, 21:3, 341–349, DOI: 10.1080/00401706.1979.10489779

[37] S.Joe Qin, Ricardo Dunia,
Determining the number of principal components for best reconstruction,
Journal of Process Control, Volume 10, Issues 2–3, 2000, Pages 245-250, ISSN 0959-1524,
https://doi.org/10.1016/S0959-1524(99)00

[38] Lu, Sixing & Lysecky, Roman. (2017). Time and Sequence Integrated Runtime Anomaly Detection for Embedded Systems. ACM Transactions on Embedded Computing Systems. 17. 1-27. 10.1145/3122785.

[39] Narkhede S. Understanding AUC-ROC Curve. Medium. Towards Data Science; 2019 [citado 10 junio 202]. Disponible en: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

[40] Matriz de confusión [Internet] [visitado 9 junio 2020]. Disponible en: https://www.interactivechaos.com/manual/tutorial-de-machine-learning/matriz-de-confusion

[41] Zelada C. Evaluación de modelos de clasificación [Internet]. RPubs. 2017 [visitado 9 junio 2020]. Disponible en: https://rpubs.com/chzelada/275494

[42] M. Sokolova, N. Japkowicz, S. SzpakowiczBeyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation Australasian Joint Conference on Artificial Intelligence, Springer (2006), pp. 1015-1021

[43] *Powers, David M W (2011).* "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation". *Journal of Machine Learning Technologies. 2 (1): 37–63.*

[44] Kim-Kwang Raymond Choo,Ali Dehgantanha Editors,Hnadbook of Big Data Privacy .Page 228

[45] Cohen J. (1960) A coefficient of agreement for nominal scales. Educ Psychol Meas 20:37-46.

[46] Sim, Julius; Wright, Chris C. (2005). "The Kappa Statistic in Reliability Studies: Use, Interpretation, and Sample Size Requirements". Physical Therapy. 85 (3): 257–268. doi:10.1093/ptj/85.3.257. ISSN 1538-6724.

[47] Van Rossum, G., & Drake, F. L. (2009). Python 3 Reference Manual. Scotts Valley, CA: CreateSpace.

[48] Beklemysheva A. Why Use Python for AI and Machine Learning? Medium. [visitado 14 junio 2020]. Disponible en: https://steelkiwi.com/blog/python-for-ai-and-machine-learning/#:~:text=Simple%20and%20consistent,developers%20to%20write%20reliable%20systems.&text=Python%20code%20is%20understandable%20by,build%20models%20for%20machine%20learning.

[49] *Anaconda Software Distribution.* Computer software. Vers. 2-2.4.0. Anaconda, Nov. 2016. Web. <https://anaconda.com>.

[50] Mathur P. What is Anaconda and Why should I bother about it? Medium. 2016 [visitado 15 junio 2020]. Disponible en: https://medium.com/pankajmathur/what-is-anaconda-and-why-should-i-bother-about-it-4744915bf3e6

[51] Zhao, Y., Nasrullah, Z. and Li, Z., 2019. PyOD: A Python Toolbox for Scalable Outlier Detection. Journal of machine learning research (JMLR), 20(96), pp.1-7.

[52] Dataman D. Anomaly Detection with PyOD! Medium. 2020 [visitado 16 junio 2020]. Disponible en: https://towardsdatascience.com/anomaly-detection-with-pyod-b523fc47db9

[53]Introducción Colab [visitado 16 junio 2020]. Disponible en: https://colab.research.google.com/notebooks/intro.ipynb

[54] Oliphant, T. E. (2006). A guide to NumPy (Vol. 1). Trelgol Publishing USA.

[55] 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):22–30, March 2011.

[56] Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. 2015. Numba: a LLVM-based Python JIT compiler. In Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC (LLVM '15). Association for Computing Machinery, New York, NY, USA, Article 7, 1–6. DOI: https://doi.org/10.1145/2833157.2833162

[57] Numba: A LLVM-based Python JIT Compiler Siu Kwan Lam Continuum Analytics Austin, Texas slam@continuum.io Antoine Pitrou Continuum Analytics Austin, Texas apitrou@continuum.io Stanley Seibert Continuum Analytics Austin, Texas sseibert@continuum.io https://dl.acm.org/doi/10.1145/2833157.2833162

[58] Stéfan van der Walt, S. Chris Colbert and Gaël Varoquaux. The NumPy Array: A Structure for Efficient Numerical Computation, Computing in Science & Engineering, 13, 22-30 (2011), DOI:10.1109/MCSE.2011.37

[59] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay. Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research, 12, 2825-2830 (2011)

[60] Wikipedia contributors, "Phrases from The Hitchhiker's Guide to the Galaxy," [visitado 28 junio 20] Wikipedia, The Free Encyclopedia, Disponible en: https://en.wikipedia.org/w/index.php?title=Phrases_from_The_Hitchhiker%27s_Guide_to_the_Galaxy&oldid=963833139