UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA DE COMPUTACIÓN

IMPLEMENTACIÓN WEB DE ALGORITMOS DE CLUSTERING CON RESTRICCIONES DE TAMAÑO

Trabajo de titulación previo a la obtención del
Título de Ingenieros en Ciencias de la Computación

AUTORES: DILAN JOHAN MAIGUA JIMENEZ
SANTIAGO PAÚL MIÑO BASANTES
TUTOR: DIEGO FERNANDO VALLEJO HUANGA

Quito - Ecuador
2024

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, Dilan Johan Maigua Jimenez con documento de identificación N.º 1726301508 y Santiago Paúl Miño Basantes con documento de identificación N.º 1751426584; manifestamos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Quito, 7 de agosto del 2024

Atentamente,

Dilan Johan Maigua Jimenez
1726301508

Santiago Paúl Miño Basantes
1751426584

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

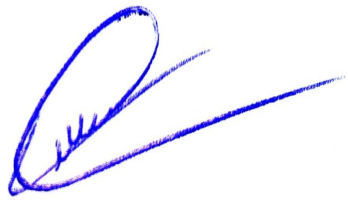
Nosotros, Dilan Johan Maigua Jimenez con documento de identificación No. 1726301508 y Santiago Paúl Miño Basantes con documento de identificaci´on No. 1751426584, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del Artículo Académico: "Implementación Web de Algoritmos de Clustering con Restricciones de Tamaño", el cual ha sido desarrollado para optar por el título de: Ingenieros en Ciencias de la Computación, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribo este documento en el momento que hago la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Quito, 7 de agosto del 2024

Atentamente,

Dilan Johan Maigua Jimenez
1726301508



Santiago Paúl Miño Basantes
1751426584

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Diego Fernando Vallejo Huanga con documento de identificación N° 1720162708, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: IMPLEMENTACIÓN WEB DE ALGORITMOS DE CLUSTERING CON RESTRICCIONES DE TAMAÑO, realizado por Dilan Johan Maigua Jimenez, con documento de identificación N.º 1726301508 y por Santiago Paúl Miño Basantes con documento de identificación N.º 1751426584, obteniendo como resultado final el trabajo de titulación bajo la opción Artículo Académico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Quito, 7 de agosto del 2024

Atentamente,

Ing. Diego Fernando Vallejo Huanga, MSc
1720162708

Implementación Web de algoritmos de Clustering con Restricciones de Tamaño

1st Dilan Johan Maigua Jimenez
dmaigua@est.ups.edu.ec

2nd Santiago Paúl Miño Basantes
sminob@est.ups.edu.ec

3rd Diego Vallejo-Huanga
dvallejoh@ups.edu.ec

***Resumen*—El *clustering* con restricciones de tamaño es una técnica en el campo del aprendizaje automático que busca agrupar datos asegurando que cada grupo cumpla con ciertos criterios predefinidos de tamaño. Este enfoque es útil para ofrecer una distribución equitativa o desigual de datos dentro de grupos específicos, lo cual es importante para ciertas aplicaciones. En este estudio, se implementaron dos algoritmos de *clustering*, SC-Medoids y CSCLP, en una aplicación web desarrollada con R Studio, el paquete Shiny y su posterior despliegue del servicio en Shinyapps.io. El proceso metodológico adoptó un enfoque en cascada, con fases secuenciales que ayudaron a estructurar el desarrollo del sitio web, generando así una aplicación que permite a los usuarios realizar *clustering* en tiempo real y ajustar parámetros de manera interactiva. Se realizaron pruebas de estrés, carga, tiempo de ejecución y uso de memoria RAM para validar la estabilidad y eficiencia del sistema bajo ciertas condiciones. Además, se evaluó la usabilidad mediante encuestas, las cuales proporcionaron *insights* sobre la experiencia de los usuarios, abordando aspectos como la facilidad de uso, navegación intuitiva y nivel de satisfacción general. Los resultados muestran un alto nivel de aceptación y recomendación de la aplicación web. Tras realizar pruebas post despliegue, se ha verificado que la página soporta hasta 50 usuarios activos simultáneamente, mientras que en las pruebas de estrés la aplicación es capaz de procesar aproximadamente 90 peticiones en alrededor de cinco segundos. En cuanto al procesamiento del algoritmo SC-Medoids soporta datasets de hasta 3500 instancias, en comparación con CSCLP, que soporta hasta 1800.**

Palabras Clave*—*SC-Medoids, CSCLP, Aplicación Web, R Shiny, Usabilidad, Experiencia de Usuario, Aprendizaje Semi-Supervisado.

***Abstract*—Size-constrained clustering is a technique in the field of machine learning that seeks to group data by ensuring that each group meets specific predefined size criteria. This approach helps provide an equal or unequal distribution of data within particular groups, which is essential for specific applications. In this study, two clustering algorithms, SC-Medoids, and CSCLP, were implemented in a web application developed with R Studio, the Shiny package, and its deployment of the service in Shinyapps.io. The methodological process adopted a waterfall model, with sequential phases that helped to structure the development of the website, thus generating an application that allows users to perform clustering in real-time and adjust parameters interactively. Stress, load, execution time, and RAM usage tests were performed to validate the stability and efficiency of the system under certain conditions. In addition, surveys evaluated usability, which provided insights into user experience, addressing aspects such as ease of use, intuitive navigation, and overall satisfaction level. The results show a high level of acceptance and recommendation of the web application. After performing post-deployment tests, it has been verified that the page supports up to 50 active users simultaneously. In contrast, the application can process approximately 90 requests in about five seconds in stress tests. As for the processing of the SC-Medoids algorithm, it supports datasets of up to 3500 instances, compared to CSCLP, which supports up to 1800.**

***Keywords*—SC-Medoids, CSCLP, Web Application, R Shiny, Usability, User Experience, Semi-Supervised Learning**

I. Introducción

Actualmente, en el contexto digital, la Experiencia del Usuario (UX) juega un papel fundamental en el éxito de un sitio web. La UX no solo comprende la interfaz visual de la plataforma, sino también la facilidad de uso, la eficiencia en la realización de tareas y la satisfacción general del individuo al interactuar con el sistema. La UX abarca todos los procesos necesarios para que los usuarios naveguen de manera armónica e interactiva. Una buena UX aumenta la satisfacción del usuario, la lealtad y el compromiso, evitando frustración, confusión y abandono del sistema [1].

Frecuentemente, los algoritmos computacionales son implementados de forma local para satisfacer una necesidad específica, lo que puede ser un obstáculo para la experiencia del usuario. La implementación de estos algoritmos en una plataforma digital no solo requiere garantizar su funcionalidad adecuada, sino también una integración fluida con la interfaz y mantener altos estándares de rendimiento y eficiencia [2].

La integración de algoritmos en plataformas web, generalmente, requiere de varios procesos adicionales en la implementación. Estos algoritmos suelen estar escritos en lenguajes de programación específicos y pueden ser difíciles de adaptar a la arquitectura de una plataforma web. Además, estos algoritmos pueden tener requisitos de rendimiento específicos que deben cumplirse para garantizar una UX fluida. Proporcionar una experiencia coherente y satisfactoria, mientras se integran los algoritmos en una plataforma web, es crucial para maximizar la utilidad y la adopción de la herramienta por parte de los usuarios [3].

Por otro lado, dentro del área de aprendizaje de máquina existe un paradigma, denominado aprendizaje no supervisado, que utiliza como entrada únicamente las variables independientes para encontrar patrones en los conjuntos de datos. Una de las técnicas más frecuentemente utilizadas es el *clustering*, que también suele ser utilizada en minería y análisis de datos, para agrupar en *clusters* a las instancias (objetos) de acuerdo a una medida de similitud específica [4]. El objetivo primario en un proceso de agrupamiento es la identificación de patrones y

estructuras en los datos, con el fin de facilitar una comprensión profunda de los fenómenos estudiados [5].

Además, el aprendizaje semi-supervisado es una técnica en el campo del aprendizaje automático, que combina elementos de los enfoques supervisado y no supervisado para mejorar la identificación de patrones. Este enfoque es aplicable a tareas como clasificación, regresión y *clustering*, integrando datos tanto etiquetados como no etiquetados. En el contexto específico del *clustering* semi-supervisado, se emplean los datos etiquetados para dirigir la formación de *clusters*, asegurando el cumplimiento de restricciones específicas [6]. Por consiguiente, el *clustering* con restricciones de tamaño impone límites mínimos o máximos en la cantidad de elementos permitidos en cada *cluster*, lo cual mejora la robustez y representación de los resultados visuales [7].

Este artículo científico se enfoca en la implementación y análisis de dos algoritmos de *clustering* con restricciones de tamaño, diseñados específicamente para abordar la segmentación de datos en un entorno de aprendizaje semi-supervisado. Los algoritmos se integrarán en un entorno web con el propósito de mejorar la usabilidad y la eficiencia del usuario final. La investigación se centra en la adaptabilidad de los algoritmos a datos dinámicos y experiencia de usuaria al momento de trabajar con estos algoritmos desplegados en la web.

A. Trabajos Relacionados

En el contexto de este estudio, se ha realizado una revisión de la literatura científica relacionada, lo que ha permitido identificar algunos trabajos que analizan el agrupamiento con restricciones de tamaño. Estos trabajos permiten contextualizar nuestra investigación y establecer relaciones, entre diferentes aproximaciones, que se están desarrollando en el campo del aprendizaje automático semi-supervisado.

Zhang et al. señalan la importancia de considerar restricciones de tamaño en el *clustering* semi-supervisado, especialmente en el contexto de conjuntos de datos desequilibrados, donde el número de instancias en las clases puede variar significativamente. Este trabajo propone la integración de restricciones de tamaño y restricciones de pares simultáneamente en un marco unificado. Además, se minimiza la diferencia entre la distribución de tamaños de clase, i.e., el *ground truth* y el resultado de la solución del algoritmo de *clustering*. Se realizaron experimentos en varios conjuntos de datos de referencia, incluyendo *datasets* equilibrados y desequilibrados, para evaluar el rendimiento del algoritmo propuesto con diferentes métricas como Información Mutua Normalizada (NMI) y la Puntuación de Alineación (AS). Cada *dataset* se utiliza para evaluar diferentes aspectos del algoritmo, como la efectividad de las restricciones de tamaño y de pares, la escalabilidad y la robustez en diferentes escenarios [8].

Presentando un enfoque para el problema de agrupamiento con restricciones de tamaño equilibrado y optimización del Error Cuadrático Medio (MSE) bajo condiciones de equilibrio, Tang et al. [9] proponen el algoritmo llamado *Balanced Clustering Algorithm (BCA)* . Este método utiliza Programación Lineal Entera (ILP) para lograr un agrupamiento balanceado. Se realizaron pruebas experimentales utilizando conjuntos de datos sintéticos y reales para evaluar el rendimiento del algoritmo en comparación con otros métodos. Los resultados del método propuesto muestran un mejor rendimiento en términos de MSE y tiempo de ejecución en comparación con los métodos BK1 y BSCK2. Las mejoras porcentuales del algoritmo son de aproximadamente un 74.1% en el MSE, para 45 iteraciones y 69.9% para 93 iteraciones. En cuanto al tiempo de ejecución, se observa una mejora de aproximadamente el 67.5% para 45 iteraciones y 53.2% para 93 iteraciones.

Los robots modulares son sistemas que se componen de numerosos módulos que se coordinan para funcionar de manera inteligente. Estos módulos son las unidades básicas, que permiten la funcionalidad y flexibilidad en este tipo de sistemas y cada uno de ellos tiene diferentes componentes y capacidades. En el estudio realizado por Bassil et al. [10], se introduce el algoritmo *SC-Clust* como una solución distribuida que utiliza el conocimiento local de los módulos para lograr un agrupamiento eficiente. Las simulaciones realizadas en *VisibleSim*, un simulador 3D de eventos discretos para robots modulares, demuestran la capacidad del algoritmo para manejar conjuntos a gran escala. El enfoque se centra en evaluar el algoritmo mediante simulaciones que abarcan diversas formas y tamaños de conjuntos de módulos, mostrando su eficiencia en el proceso de auto-reconfiguración. Además, se realiza una comparación detallada con otros enfoques existentes, resaltando las ventajas y capacidades distintivas del *SC-Clust*. También se exploran aplicaciones potenciales, como la mejora y el desarrollo de algoritmos basados en grupos, resaltando sus ventajas y capacidades distintivas.

Ganganath et al. [11], proponen una modificación al algoritmo *k-means* que incorpora restricciones de tamaño para cada *cluster* de manera individual. Esta aproximación resulta adecuada para aplicaciones que requieren la formación de *clusters* con tamaños específicos. En las simulaciones con *datasets* multidimensionales, se compararon los resultados del algoritmo *k-means* estándar con el algoritmo modificado, demostrando que este último produce resultados más estables y evita mínimos locales. La inicialización selectiva de centroides, dentro de las dos simulaciones experimentales, en el algoritmo modificado mejora la consistencia en la formación de *clusters* de tamaños específicos.

Zhu et al. destacan la importancia de incorporar restricciones de tamaño en algoritmos de agrupamiento tradicionales para mejorar la precisión del agrupamiento y evitar la formación de *clusters* atípicos. Proponen un algoritmo heurístico que convierte problemas de agrupamiento con restricciones de tamaño en problemas de programación lineal entera, ofreciendo un enfoque para manejar restricciones de rango de tamaño aproximado en lugar de restricciones exactas de tamaño de *cluster*. Para evaluar la efectividad de este enfoque heurístico, utilizaron cuatro conjuntos de datos, midiendo la calidad del agrupamiento con métricas como la exactitud, el Índice Aleatorio Ajustado (ARI), NMI y la entropía. El algoritmo fue evaluado con varios *datasets* de prueba y de-

mostró su capacidad de generar *clusters* con rangos de tamaño aproximados [12].

Vallejo et al. [13] se enfocan en la aplicación de algoritmos de *clustering* semi-supervisado para el agrupamiento de artículos científicos en conferencias. En su estudio, proponen dos algoritmos, SC-Medoids y *Clustering with Size Constraints with Linear Programming* (CSCLP), que pueden formar grupos de documentos cumpliendo con restricciones específicas de tamaño. Utilizan técnicas de procesamiento de lenguaje natural y modelos de recuperación de información para crear matrices de disimilitud que guían a los algoritmos de agrupamiento. Los resultados obtenidos muestran la efectividad de los algoritmos propuestos en la formación de grupos que cumplen con restricciones de tamaño en diferentes conjuntos de datos y evaluados con índices de validación interna y externa.

Este artículo científico se estructura en cuatro secciones. En la Sección II, se detalla la metodología empleada, incluyendo la descripción de los *datasets*, los algoritmos seleccionados para el proceso de *clustering*, la metodología de evaluación de la UX y la interfaz web desarrollada para garantizar una navegación eficiente y el aprovechamiento óptimo de todas las funcionalidades disponibles. La Sección III presenta los resultados obtenidos, con un análisis enfocado en los tiempos de ejecución de los *datasets* y uso de memoria RAM, con ambos algoritmos. Además, se analiza el comportamiento de la aplicación web mediante pruebas de estrés, carga y usabilidad. Finalmente, en la sección IV, se discuten las conclusiones del estudio, analizando las implicaciones de los resultados y proponiendo posibles direcciones para futuras investigaciones.

II. Metodología

La Figura 1 esquematiza el proceso metodológico, de tipo cascada, utilizado en esta investigación. Para la primera etapa, se realizó una búsqueda de varios conjuntos de datos de diversas áreas, los cuales se utilizarán para validar el funcionamiento y evaluar el rendimiento de los algoritmos de agrupamiento SC-Medoids y CSCLP. Estos conjuntos de datos fueron seleccionados para asegurar una amplia variedad de características y tamaños. Luego, se procedió a implementar los algoritmos en el lenguaje de programación R, bajo el entorno de desarrollo *R Studio*.

Posteriormente, se llevaron a cabo pruebas en un entorno local utilizando todos los conjuntos de datos seleccionados. Estas pruebas tienen como objetivo verificar la funcionalidad de los algoritmos, analizando los resultados para asegurar que cumplieran con las restricciones de tamaño impuestas. Con los algoritmos validados en el entorno local, se inició el desarrollo de la interfaz de usuario (UI) y el servidor (Server) de la aplicación web utilizando el paquete *Shiny* de R. Esta aplicación web utiliza la arquitectura cliente-servidor, donde el servidor centraliza todas las operaciones y recursos necesarios para el funcionamiento. Mientras que el cliente se conecta al servidor para obtener y representar datos, así como para realizar acciones que afecten el estado del servidor. En este modelo, los clientes actúan principalmente como interfaces de usuario, delegando al servidor las tareas complejas y el procesamiento intensivo [14].

Para esta investigación la aplicación web gestiona la carga de archivos y la configuración de parámetros para el agrupamiento. Además, la aplicación almacena y utiliza los datos necesarios para el análisis y presentación de resultados. Dado que la interfaz de usuario, es una *Single Page Application*

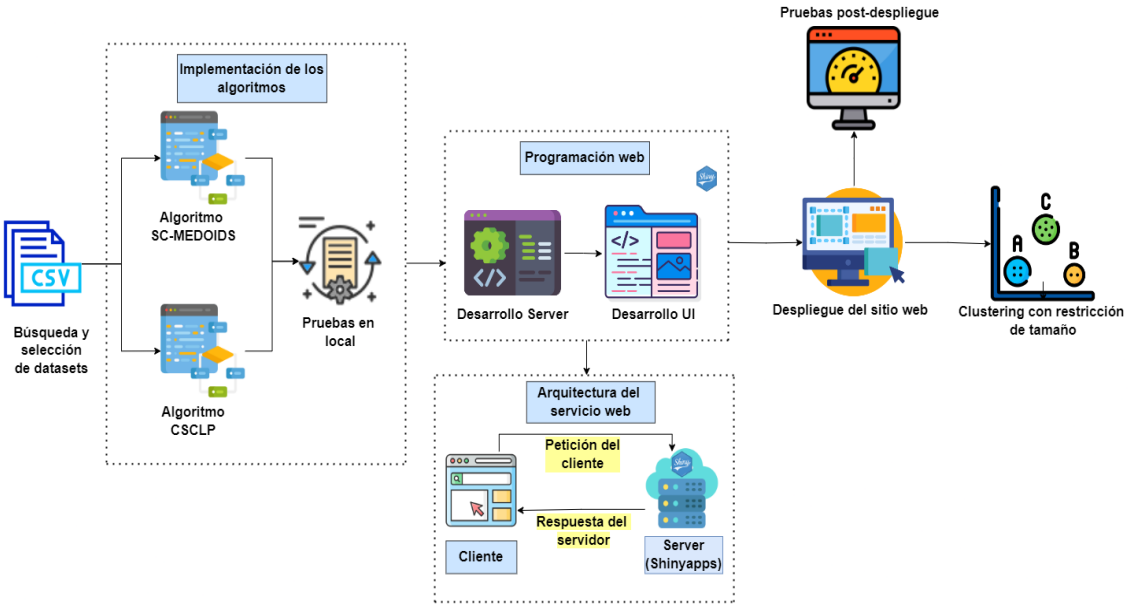


Fig. 1: Diagrama del proceso metodológico para la implementación y despliegue de algoritmos de *clustering* con restricciones de tamaño en la aplicación web

(SPA), entonces, la experiencia es fluida e interactiva. Los resultados del agrupamiento se calculan en tiempo real según la configuración del usuario y se presentan dinámicamente en la misma página, sin necesidad de recargarla.

La página es de acceso público y sus servicios se pueden consumir en https://clustering-algorithms-with-size-constraints.shinyapps.io/shinyapps/

Con el objetivo de garantizar la funcionalidad óptima de la aplicación web, posterior al despliegue, se llevó a cabo una fase de pruebas. Para esto se utilizaron pruebas de estrés, carga y de usabilidad que permite evaluar y mejorar la UX. Las pruebas de estrés buscan evaluar la resistencia de la aplicación ante acontecimientos extremos y determinar su estabilidad [15]. Por otro lado las pruebas de carga son fundamentales para asegurar que el sistema funcione de manera óptima bajo condiciones reales de uso con varios usuarios conectados similtaneamenteutiliza [16]. Al simular diferentes escenarios de carga, como picos de usuarios o incrementos en el volumen de datos, podemos identificar cómo nuestro sistema responde y dónde podrían surgir problemas de rendimiento. Esto nos permite ajustar los recursos y optimizar la capacidad de respuesta del sistema, asegurando una experiencia fluida para los usuarios incluso durante períodos de alta demanda. Para la implementación de ambas pruebas se utilizó la aplicación *open-source JMeter*. Esta herramienta no solo posibilitó la simulación de múltiples usuarios simultáneos y la recreación de situaciones extremas para identificar puntos débiles y medir la capacidad del sistema, sino que también su configuración flexible permitió ajustar el número de usuarios y la frecuencia de las solicitudes. Esto facilitó la evaluación del rendimiento bajo condiciones específicas y la detección de posibles cuellos de botella [17].

Además, las pruebas de usabilidad se enfocan en la experiencia y comentarios de los usuarios al utilizar el sitio web. Estas pruebas registran su comportamiento y procesos cognitivos para comprender su confort en la aplicación. Se utilizó *Google Forms* para llevar a cabo una encuesta de usabilidad, recogiendo comentarios y evaluando la UI desde la perspectiva de los usuarios finales. Este método permite identificar posible mejoras y la optimización continua del diseño web [18].

A. Descripción de los Conjuntos de Datos

Para esta investigación se emplearon 15 *datasets* estructurados. La Tabla I presenta su clasificación según el área de conocimiento definida por la UNESCO e identificados por un número de ID, nombre del *dataset*, número de variables y clasificados por una taxonomía empírica definida en función del número de instancias contenidas. Estos *datasets* fueron utilizados para evaluar el rendimiento de los algoritmos de *clustering* SC-Medoids y CSCLP, proporcionando un conjunto diverso para el análisis comparativo.

El objetivo de clasificar a los *datasets*, en esta investigación, es conocer el comportamiento de los algoritmos para datos de diferente naturaleza y volumen. De acuerdo a esta taxonomía los conjuntos de datos pueden ser: *i) Pequeños*, si su número instancias es de hasta 2000. Esta cantidad restringida de datos permite ejecutar pruebas rápidas, ideales para análisis exploratorios iniciales. *ii) Medianos*, con un número de instancias entre 2000 y 4000. Aunque aún son manejables en términos de volumen, ofrecen una mayor cantidad de información para realizar pruebas más detalladas. *iii) Grandes* con un número de instancias entre 4000 y 6500. Estos conjuntos de datos pueden proporcionar *insights* más profundos sobre el comportamiento de los algoritmos en situaciones de alta dimensionalidad y tamaño.

Tabla I: Descripción de los metadatos para los *datasets* estructurados de prueba

ID	Nombre	Instancias	Variables	Área de Conocimiento	Taxonomía
1	Iris	150	3	Ciencias Naturales	Pequeño
2	Heart Disease	1025	14	Ciencias de la Salud	Pequeño
3	Obesity Levels	2111	17	Ciencias de la Salud	Pequeño
4	Glass Identification	214	9	Ciencias Naturales	Pequeño
5	Breast Cancer Wisconsin	568	30	Ciencias de la Salud	Pequeño
6	Engineering Graduate Salary	2998	34	Finanzas	Mediano
7	Water Probability	3276	10	Ciencias Naturales	Mediano
8	Cure The Princess	2338	14	Fantasía	Mediano
9	AIDS Clinical	2139	24	Ciencias de la Salud	Mediano
10	Migration from Mexico to USA data	2443	10	Ciencias Sociales	Mediano
11	Bank Loan Approval	5000	14	Finanzas	Grande
12	Wine Quality	6497	13	Enología	Grande
13	Clustering of cycling	4435	11	Análisis Deportivo	Grande
14	Turkiye-student-evaluation	5820	33	Ciencias Matemáticas	Grande
15	Abalone UCI	4177	8	Ciencias Naturales	Grande

La Tabla II presenta información adicional de los *datasets*, de los tamaños y número de grupos, de acuerdo al *(ground truth)* de cada conjunto de datos de prueba. Esta información permite que los algoritmos conozcan de antemano, tanto el número de grupos esperados, como la cardinalidad de cada uno.

Tabla II: Metadatos de cardinalidad y número de grupos de los *datasets.*

ID	Tamaños de los grupos	Número de grupos
1	50, 50, 50	3
2	499, 526	2
3	272, 287, 351, 297, 324, 290, 290	7
4	70, 76, 17, 13, 9, 29	6
5	356, 212	2
6	226, 2172	2
7	1998, 1278	2
8	1177, 1161	2
9	1618, 521	2
10	330, 593, 392, 93, 162, 873	6
11	3530, 1470	2
12	30, 216, 2138, 2836, 1079, 193, 5	7
13	1399, 312, 467, 356, 290, 549, 503, 185, 374	9
14	775, 1444, 3601	3
15	1307, 1342, 128	3

B. Descripción de los Algoritmos de Clustering

Se utilizó dos algoritmos de *clustering* semi-supervisado para esta investigación: SC-Medoids y CSCLP [13]. SC-Medoids es una variante del algoritmo k-medoids que incorpora información de restricciones de pares para mejorar la selección de medoids y la asignación de datos, lo que resulta en *clusters* más coherentes. Por otro lado, CSCLP combina técnicas de *clustering* con programación lineal para manejar

restricciones de pares, construyendo un grafo de similitud que incorpora estas restricciones para la asignación final de *clusters*.

1) SC-Medoids: El algoritmo SC-Medoids identifica centroides dentro del conjuntos de datos a partir de una matriz de distancias. El pseudocódigo del algoritmo SC-Medoids 1 muestra el funcionamiento de este proceso de agrupamiento con restricciones de tamaño.

Algoritmo 1 Pseudocódigo de SC-Medoids

Entrada: D, k, E, C
Salida: R_l
Inicialización:
if C es nulo **then**
    Seleccionar k instancias aleatorias de D como medoides iniciales y asignarlos a C
end if
Asignar cada instancia al cluster del medoide más cercano
Ordenar las instancias según su mínima distancia a cualquier medoide
for cada cluster i de 1 a k **do**
    Asignar las primeras $E[i]$ instancias más cercanas al cluster i
    Eliminar estas instancias de la lista de instancias ordenadas
end for
for cada instancia restante en la lista ordenada **do**
    Asignar la instancia al cluster del medoide más cercano
end for
Crear un vector R_l de etiquetas con la asignación final de cada instancia a un cluster
for cada cluster i de 1 a k **do**
    Asignar la etiqueta a todos las instancias del cluster
end for
Devolver: R_l

SC-Medoids, recibe como entrada una matriz de similitudes D y particiona los datos en k *clusters* utilizando medoides como representantes. Cada instancia se asigna al medoide más cercano basado en D. Para cada clúster, el algoritmo busca una nueva instancia que minimice la suma de las distancias a todas las demás instancias en el mismo clúster, convirtiéndose en el nuevo medoide. El vector E define las restricciones de tamaño de cada *cluster*. Si C está vacío, se seleccionan aleatoriamente los medoides iniciales. Este proceso de asignación y actualización se repite hasta que los medoides no cambian. Los resultados se almacenan en un vector R_l de etiquetas, donde cada elemento indica el grupo al que pertenece cada instancia. A diferencia del algoritmo *k-means*, que calcula centroides como puntos medios en el espacio de características, este algoritmo siempre selecciona a las instancias como representaciones de los grupos. Esto es útil cuando los *datasets* representan información no estructurada.

2) CSCLP: El algoritmo CSCLP utiliza una matriz de distancias y programación lineal para realizar el agrupamiento con restricciones. El pseudocódigo del algoritmo CSCLP 1 muestra el funcionamiento de este proceso de agrupamiento con restricciones de tamaño.

Algoritmo 2 Pseudocódigo de CSCLP

Entrada: X, D, k, f, Aeq, beq, E
Salida: R_l
Método:
Realizar clustering inicial usando k-means sobre X para obtener D
$r \leftarrow$ número de elementos en D
Verificar si los clusters cumplen con las restricciones de tamaño
if cumplen las restricciones **then**
    Devolver los clusters obtenidos con k-means
else
    Eliminar todas las filas de D excepto las de los centroides
    Resolver el LP minimizando f con restricciones Aeq, beq, variables binarias
    Construir matriz $s1$ de dimensiones $k \times r$ usando D
    Crear matriz st de dimensiones $k \times r$, inicializada en ceros
    Definir $result$ con las soluciones óptimas del LP
    Configurar columnas de st como diagonal de tamaño k
    Asignar valores de $s1$ a columnas de st
    Inicializar vector y de longitud r
    Para cada i de 1 a k:
        Asignar i a posiciones de y donde $st[i,]$ es 1
    Devolver R_l
end if

CSCLP se utiliza para agrupar un conjunto de datos X en k *clusters*, comenzando con un *clustering* inicial mediante k-means que genera una matriz de similitudes D, reflejando las distancias entre los centroides e instancias de los *clusters*. Luego, se verifica si cumplen con las restricciones de tamaño específicas, en caso contrario, se aplica un proceso de programación lineal lp donde se minimiza una función objetivo f bajo restricciones de igualdad Aeq y beq, asegurando que todas las variables sean binarias. Este paso optimiza la distribución de instancias en los *clusters*, utilizando la información contenida en D para construir matrices $s1$ y st que facilitan la asignación final de *clusters*. El resultado del lp proporciona etiquetas R_l que indican a qué cluster pertenece cada instancia, optimizando tanto la estructura de los *clusters* como la conformidad con las restricciones de tamaño establecidas.

C. Descripción de la Metodología de Desarrollo Web

La implementación del *backend* de la aplicación en Shiny usa una estructura modular y reactiva que segmenta el sistema en partes autónomas y manejables llamadas módulos. La reactividad permite al sistema responder rápida y eficientemente a eventos y cambios. Cada módulo encapsula su propia lógica y

datos, promoviendo la reusabilidad y la mantenibilidad. Esta combinación facilita la comunicación basada en eventos entre los módulos, mejorando la tolerancia a fallos y permitiendo que el sistema esté organizado, sea flexible y adaptativo a diferentes cargas y escenarios [19]. La lógica del servidor está diseñada para responder a eventos específicos del usuario y manipular los datos donde se aplican los algoritmos de agrupación con restricciones de tamaño. El proceso comienza con la carga y preparación de los datos, en formato *CSV*, subidos por el usuario. Luego, se calculan matrices de similitues utilizando dos métricas, el coseno de Salton y la distancia euclidiana, según elija el usuario. Estas matrices son las entradas para ambos algoritmos de *clustering*. Para manejar las interacciones del usuario, se utilizan funciones que responden automáticamente a acciones que especifiquen las restricciones de tamaño. Estas ayudan a verificar que los parámetros ingresados sean válidos antes de continuar con el proceso algorítmico de *clustering*. Los resultados del agrupamiento se presentan en tablas interactivas, que permiten al usuario explorar y analizar los grupos identificados, junto con las características de los datos originales. Se genera un gráfico dinámico mediante el Análisis de Componentes Principales (PCA), que ayuda a interpretar la estructura generada de los grupos obtenidos.

Por otro lado, el *frontend* de la aplicación en *Shiny* está diseñado para facilitar la interacción del usuario con los algoritmos de *clustering*. El UI incluye un panel lateral que solicita el ingreso de los datos junto a las restricciones de tamaño que necesitan los algoritmos. La interfaz principal se divide en dos pestañas, donde, la primera permite ver la información del conjunto de datos ingresado y seleccionar los parámetros para el proceso de agrupamiento, como se muestra en la Figura 2.

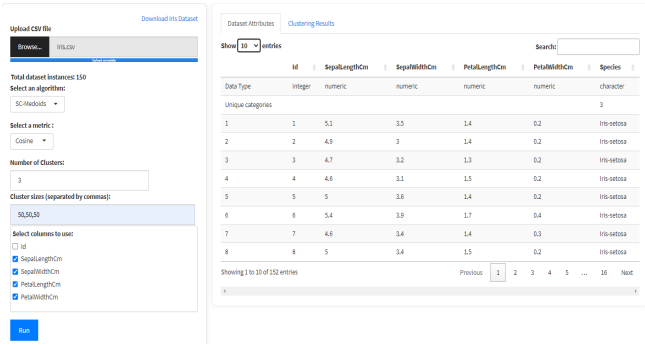


Fig. 2: Selección de parámetros y visualización del *dataset* en el UI

La segunda pestaña muestra los resultados del proceso de *clustering* de manera gráfica, mediante PCA, como se observa en la Figura 3. Las instancias agrupadas, con su respectivo identificador de *cluster*, comparado con el agrupamiento ideal *ground truth* se muestran en una tabla estructurada.

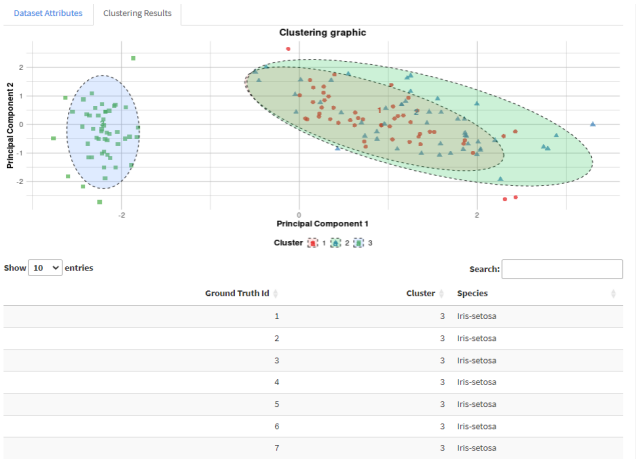


Fig. 3: Pestaña de resultados en la UI después de un proceso de *clustering*.

D. Metodología de Evaluación de la UX

En la evaluación de la usabilidad web, las encuestas permiten recopilar percepciones detalladas de usuarios con cierta experiencia informática. Para este estudio estas evaluaciones fueron realizadas a través de *Google Forms*, permitiendo dirigir la encuesta a individuos familiarizados con las TICs. Estos usuarios proporcionaron *insights* sobre la facilidad de uso, la navegación intuitiva y la claridad de la interfaz de usuario. Además, los participantes evaluaron aspectos técnicos como el tiempo de carga, el rendimiento de la aplicación y la comprensibilidad de los resultados de *clustering*. La Tabla III muestra las preguntas utilizadas para recopilar respuestas de los usuarios sobre la aplicación web.

ID	Pregunta
1	Selecciona tu género
2	Escoge tu edad
3	Grado de instrucción formal
4	¿Qué tan fácil fue entender cómo usar la aplicación?
5	¿Cómo calificarías la facilidad de navegación de la aplicación?
6	¿Encuentras que la interfaz de usuario es intuitiva?
7	¿Cuál de las siguientes funcionalidades de la aplicación encontraste confusa o difícil de usar?
8	¿Qué tan rápido fue el tiempo de carga de la aplicación y las respuestas de la interfaz?
9	¿Experimentaste algún problema de rendimiento durante el uso de la aplicación?
10	¿Fue clara y comprensible la presentación de los resultados de clustering?
11	¿Cómo calificarías tu satisfacción general con la aplicación?
12	¿Recomendarías esta aplicación a otros usuarios interesados en análisis de datos y clustering?

Tabla III: Preguntas realizadas para medir la usabilidad de la pagina web en base a encuestas.

Las preguntas de la encuesta abarcan diversos aspectos, incluyendo el perfil demográfico del usuario (género, edad y

nivel educativo), facilidad de uso y navegación en la aplicación, intuición de la interfaz de usuario, percepción del rendimiento técnico (tiempo de carga, velocidad de respuesta y problemas de rendimiento), comprensibilidad de los resultados del análisis (*clustering*), satisfacción general con la aplicación y la disposición a recomendarla a otros interesados en el análisis de datos. Esta metodología facilita la identificación de áreas de mejora y la optimización del diseño para satisfacer las necesidades específicas de los usuarios informáticos. La encuesta de usabilidad empleada puede ser consultada en la siguiente URL: https://forms.gle/7Yb4mksTTXHsXeqN8.

E. Metodología de Evaluación del Sistema Web

Después del despliegue de la aplicación, los dos algoritmos fueron evaluados para verificar su funcionalidad en la web. El objetivo es verificar el desempeño de la página y para esto se incluyeron mediciones de velocidad, picos de consumo de RAM y eficiencia, bajo diferentes condiciones de carga y estrés.

1) Pruebas de Carga: Para la ejecución de pruebas de carga se utilizó la herramienta de código fuente abierto *JMeter* y se configuraron diferentes escenarios para evaluar el rendimiento de la aplicación web. Se establecieron tres configuraciones distintas con 30, 50 y 70 hilos, respectivamente. Cada hilo se configuró para iniciar simultáneamente con un período de subida de 0 segundos, simulando así una carga instantánea máxima sobre el servidor, desde el inicio de la prueba. Además, cada hilo se ejecutó en una iteración, enviando una única solicitud durante la prueba. Esta configuración específica se seleccionó para analizar a la aplicación con diferentes niveles de carga bajo condiciones controladas.

2) Pruebas de Estrés: Las pruebas de estrés fueron configuradas con un escenario de 10 hilos, para simular la carga de usuarios concurrentes. Esta cantidad de hilos permite controlar el número de usuarios activos simultáneamente y evaluar la capacidad del sistema bajo condiciones de carga máxima. El período de subida de 5 segundos permite una activación gradual de los hilos, imitando así un aumento progresivo de la carga sobre el servidor, que simula el comportamiento de una aplicación durante picos de demanda repentinos. Las pruebas se ejecutan en un bucle sin fin para evaluar de forma continua el rendimiento del sistema a lo largo del tiempo, proporcionando *insights* sobre la estabilidad y capacidad de recuperación bajo condiciones de estrés prolongadas.

III. Resultados y Discusiones

Durante las pruebas realizadas en el entorno de desarrollo, se observaron varios factores que influyeron en el rendimiento del sistema, la complejidad computacional de los algoritmos de *clustering* utilizados y el volumen de datos procesados. Al desplegar la aplicación en la plataforma web, se hicieron evidentes las limitaciones adicionales debido a las restricciones de recursos de Shinyapps.io, que limitan el uso de RAM a 1GB. Inicialmente, se evaluó el algoritmo SC-Medoids, el cual demostró una eficiencia notable en términos de tiempo de procesamiento y uso de recursos, manteniendo un rendimiento aceptable incluso con conjuntos de datos considerables en pruebas locales. Sin embargo, al intentar desplegar la aplicación y realizar pruebas bajo carga en la plataforma gratuita Shinyapps.io, se evidenció que el procesamiento de *datasets* grandes excedía los recursos disponibles, resultando en caídas del sitio web. Adicionalmente, la implementación del algoritmo CSCLP presentó desafíos significativos debido a su alta complejidad computacional, lo cual se tradujo en tiempos de respuesta prolongados y una mayor incidencia de fallos en comparación con SC-Medoids, especialmente en el entorno de despliegue en producción.

A. Evaluación del Sistema en Tiempo y Memoria

Para entender cómo varía el tiempo y la RAM utilizada en el conjunto de dataset diversos, se ejecuto ambos algoritmos en ambiente local.

Los resultados obtenidos del análisis comparativo entre ambos algoritmos para diferentes distancias se presentan en la Tabla IV. La tabla muestra una comparación detallada del tiempo de ejecución y el uso de memoria RAM entre los algoritmos SC-Medoids y CSCLP para diferentes combinaciones de ID y tipos de distancia (Coseno y Euclidiana). Cada fila representa una prueba única, registrando los valores de tiempo y RAM utilizados durante la ejecución de los algoritmos. Los valores "-" indican condiciones particulares donde no se pudo obtener respuesta en los conjuntos de datos utilizados: en el caso del ID=12, la ausencia de datos se debe a que el tamaño del *dataset* era considerablemente grande, lo que excedió las capacidades de memoria disponibles. Para el ID=7, la presencia de "-" se debe a la existencia de datos faltantes (NA) en el conjunto de datos, lo cual impidió la ejecución de las pruebas correspondientes para el algoritmo CSCLP. Durante las pruebas en el entorno de desarrollo, el algoritmo SC-Medoids mostró una eficiencia notable en términos de tiempo de procesamiento y uso de recursos, manteniendo un rendimiento aceptable incluso con conjuntos de datos de tamaño considerable. No obstante, el procesamiento de *datasets* grandes excedió la capacidad ofrecida por la versión gratuita de *Shinyapps.io*, lo que resultó en caídas del sitio web debido a la limitación de recursos. Además, la implementación del algoritmo CSCLP presentó desafíos adicionales, su alta complejidad computacional, combinada con las restricciones de recursos, causó tiempos de respuesta prolongados y una mayor incidencia de caídas en la aplicación en comparación con el algoritmo SC-Medoids. En base a los resultados obtenidos para el caso del algoritmo SC-Medoids podremos obtener respuestas mientras el tamaño del conjunto de datos sea menor a 3500 instancias con un tiempo de ejecución rápido, mientras que el algoritmo CSCLP al tener un mayor coste computacional, soporta un máximo de 1800 instancias y un mayor tiempo de ejecución a comparación de SC-Medoids.

B. Resultados de las Pruebas de Carga

A partir de las configuraciones propuestas para las pruebas de carga, podemos visualizar los siguientes resultados.

ID	Distancia	Tiempo [s]		RAM peak (MB)	
		SC-Medoids	CSCLP	SC-Medoids	CSCLP
1	Coseno	0,023	0,011	233.2	239.4
	Euclidiana	0,028	0,018	235.4	244.5
2	Coseno	0,055	0,492	256.6	272.6
	Euclidiana	0,256	0,719	256.6	364.4
3	Coseno	0,642	2,04	365.8	1,022.1
	Euclidiana	0,438	1,624	954.4	1,030.9
4	Coseno	0,026	0,046	758.1	767.8
	Euclidiana	0,003	0,307	759.7	760.1
5	Coseno	0,049	0,077	761.1	761.5
	Euclidiana	0,048	0,12	774.2	762.1
6	Coseno	1,398	2,366	652.4	2,084.5
	Euclidiana	1,425	2,386	588.4	2,079.8
7	Coseno	0,644	–	535.3	–
	Euclidiana	0,863	–	674.6	–
8	Coseno	0,317	1,281	374.8	1,126.6
	Euclidiana	0,32	1,22	877.3	1,105.2
9	Coseno	0,35	0,945	657.2	1,042.7
	Euclidiana	0,292	0,984	690.7	1,068.8
10	Coseno	0,315	4,431	914.3	1,187.2
	Euclidiana	0,326	4,606	914.5	1,187.3
11	Coseno	1,604	6,056	1,053.6	2,521.8
	Euclidiana	1,22	6,249	1,044.9	2,513.9
12	Coseno	2,988	–	3,083.1	–
	Euclidiana	2,152	–	1,857.5	–
13	Coseno	1,254	32,319	2,048.2	2,935.7
	Euclidiana	0,976	36,437	1,118.8	2,932.0
14	Coseno	3,33	16,729	3,257.9	5,688.0
	Euclidiana	2,956	15,091	1,960.6	5,777.4
15	Coseno	0,851	6,695	4,111.9	6,911.5
	Euclidiana	0,821	6,391	4,112.8	6,913.3

Tabla IV: Comparación de tiempos de ejecución y RAM peak entre los algoritmos SC -Medoids y CSCLP

Basándonos en los datos registrados, podemos observar cómo el rendimiento y la tasa de errores de la aplicación web varían con el número de hilos (usuarios concurrentes) utilizados durante la prueba de carga.

Con 30 hilos, la aplicación maneja las solicitudes de manera eficiente sin errores y con un rendimiento de 9.9 solicitudes por segundo. Al aumentar a 50 hilos, se observa una ligera degradación con un 2% de errores, pero el rendimiento casi se duplica a 20.7 solicitudes por segundo, lo cual indica una buena capacidad de escalado inicial. Sin embargo, al incrementar a 70 hilos, la tasa de errores se dispara a 28.57%, lo que sugiere que la aplicación está superando su capacidad de manejo de carga, resultando en un rendimiento de 47.6 solicitudes por segundo pero con una alta tasa de fallos.

Estos resultados indican que mientras la aplicación puede escalar razonablemente bien hasta 50 usuarios concurrentes, empieza a fallar significativamente con 70 usuarios. Este límite está determinado por la capacidad del servidor para manejar solicitudes y procesar datos dentro de los parámetros definidos durante las pruebas de carga.

C. Resultados de las Pruebas de Estrés

Se realizaron las pruebas de estrés inicialmente con 10 hilos, un período de subida de 5 segundos y en un bucle sin fin, configuraciones que se ingresaron, se obtiene que con un total de 132 hilos ejecutados. Mediante la visualización de la Figura 5, se identificaron múltiples hilos que experimentaron errores

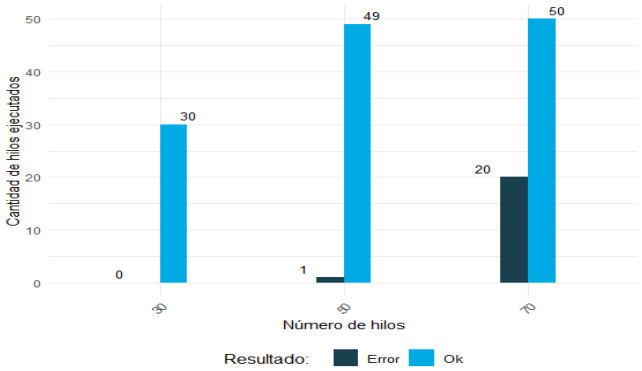


Fig. 4: Resultados de la prueba de carga con diferentes números de hilos, con la cantidad de peticiones HTTP erróneas y aceptadas

consistentes, marcados con el código de estado “Error 503 - Servicio no disponible”. Estos hilos fallidos fueron desde el rango del número 91 al 132. Estos errores comenzaron a manifestarse después del hilo 90, donde inicialmente todos los hilos operaban correctamente. Este comportamiento puede explicarse por la carga gradual que experimentó el servidor con el incremento de hilos y solicitudes. A medida que más hilos se activaron y las solicitudes aumentaron simultáneamente, el servidor alcanzó su capacidad máxima para manejar conexiones entrantes y procesar solicitudes de manera efectiva. Esto llevó a que algunos hilos recibieran respuestas de error 503, indicando que el servicio no estaba disponible para atender esas solicitudes en ese momento. Estos resultados demuestran que, a pesar de los errores identificados, la página web mantiene una capacidad de tolerancia considerable antes de alcanzar su límite máximo. En el contexto de nuestro caso de estudio, la aparición de errores a partir del hilo 91 proporciona una referencia clara del umbral de carga que el servidor puede manejar antes de degradar su rendimiento. Este umbral es crucial para planificar futuras expansiones y optimizaciones del sistema, asegurando que se puedan implementar mejoras específicas para aumentar la capacidad de procesamiento y la resiliencia del servidor.

D. Evaluación de Usabilidad del UI

Las preguntas de las sección II-D nos ayudaron a comprender como los usuarios se familiarizaron con el UI y facilidad a trabajar el UX.

Empezando por las primeras 25 respuestas recibidas para evaluar la UX, estas se demostraron mediante el uso de gráficos para obtener una mejor comprensión de los registros analizados. En la Figura 6, se presenta un gráfico tipo piramidal que compara el género con el rango de edad de los encuestados, con el objetivo de visualizar de manera clara y comparativa cómo se distribuyen estas variables demográficas en la muestra analizada.

En la Figura 7, se presenta la relación entre el grado de instrucción formal de los usuarios y su percepción de la facilidad de uso de la aplicación. Los grados académicos se

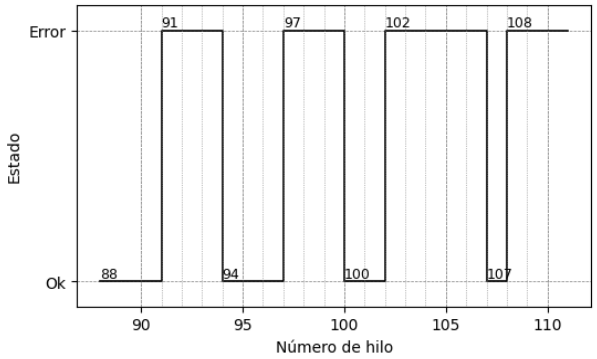


Fig. 5: Resultados de la prueba de estrés en el rango de hilos (88 a 111) donde se encontró transiciones entre peticiones aceptadas y rechazadas

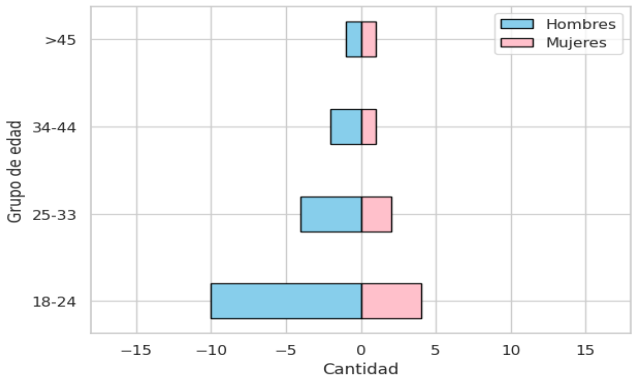


Fig. 6: Resultado gráfico en base al rango de edades con genero del encuestado

clasificaron en varias categorías: estudios universitarios en curso, estudios de posgrado, título universitario y estudios universitarios inconclusos. La percepción de la facilidad de uso se evaluó en una escala que incluye términos como “Muy fácil”, “Fácil”, “Neutral” y “Difícil”.

Los resultados indican que los usuarios con estudios universitarios en curso y aquellos con título universitario reportan niveles altos de facilidad de uso, predominando las respuestas positivas. En contraste, los usuarios con estudios universitarios inconclusos o títulos de posgrado muestran percepciones más variadas, que abarcan desde respuestas neutrales hasta positivas. Este patrón sugiere una correlación entre el nivel educativo y la experiencia percibida de uso de la aplicación, destacando la importancia de considerar la formación académica al momento de diseñar las interfaces de una aplicación web.

El gráfico de araña representado en la Figura 8 revela la variada percepción de los usuarios en tres aspectos importantes de la aplicación: satisfacción, recomendación y velocidad de página. Se utilizaron las siguientes métricas: “Muy mala”, “Mala”, “Regular”, “Buena” y Muy buena”. La mayoría de las respuestas tienden a ser “Muy buena” y “Buena”. Estos resultados indican una percepción general positiva de la aplicación entre los usuarios. La alta puntuación en satisfacción

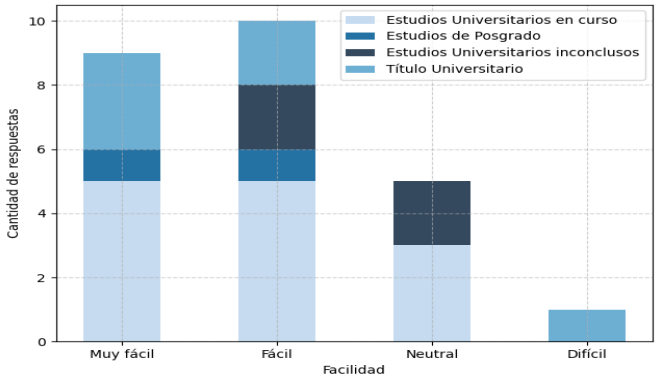


Fig. 7: Gráfico de barras apiladas en base a al grado de instrucción formal del encuestado y su percepción de facilidad de uso de la pagina web

sugiere que la mayoría de los usuarios están contentos con su experiencia. Asimismo, la disposición a recomendar la aplicación refleja un alto nivel de confianza y satisfacción general. La percepción positiva de la velocidad de página destaca la eficiencia del rendimiento técnico de la aplicación, lo cual es crucial para mantener una experiencia de usuario fluida y agradable.

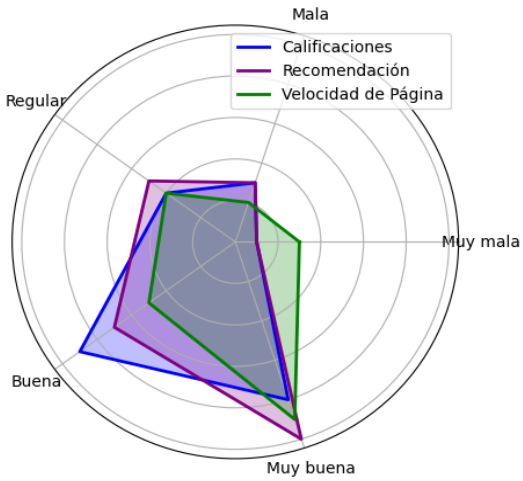


Fig. 8: Gráfico de araña de la percepción de los usuarios en cuanto a nivel de satisfacción, recomendación y velocidad de la pagina.

IV. Conclusiones

La implementación de los algoritmos SC-Medoids y CSCLP en una aplicación web desarrollada con R Studio y Shiny ha demostrado ser efectiva para el análisis de datos con restricciones de tamaño en los *clusters*. Esta herramienta proporciona a los usuarios la capacidad de ajustar varios parámetros, mejorando así la accesibilidad y usabilidad de la aplicación. En las pruebas de estrés la página web alcanzó un 25% de tasa de errores y un rendimiento de 22.5 solicitudes por segundo, en base a 130 hilos ejecutados. Por otro lado, la prueba de

carga demuestra que la página puede soportar un promedio de 50 usuarios conectados simultáneamente. Una vez que se supera esta capacidad, el sistema empezará a rechazar nuevas conexiones de usuarios debido a que alcanza su límite de carga máximo.

Ofrecer una herramienta con visualizaciones claras y dinámicas de los resultados, como tablas interactivas y gráficos, facilitan la interpretación y el análisis de los datos agrupados, proporcionando *insights* para la toma de decisiones en diversos contextos. En cuanto al desarrollo de la interfaz de usuario (UI) y la experiencia de usuario (UX) en la aplicación Shiny, mediante la encuesta realizada, los resultados indican que el diseño es intuitivo y facilita la interacción del usuario. La presentación estructurada de los resultados de *clustering*, y una navegación clara, contribuyeron a mejorar la experiencia visual del usuario. Los resultados de usabilidad obtenidos a través de encuestas dirigidas a un publico objetivo, reflejan una percepción positiva de la aplicación web en general. En cuanto a la ejecución de los algoritmos, la limitación de 1 GB de RAM en el entorno de Shinyapps afecta la capacidad de procesamiento de conjuntos de datos grandes en la aplicación. Esta limitación de RAM debe ser abordada para garantizar un rendimiento óptimo y escalabilidad en escenarios con *datasets* de alta dimensionalidad y tamaño.

Como prospectiva también se sugiere aumentar el número de encuestados para obtener una muestra más representativa, mejorando la precisión de los resultados y proporcionando una visión más detallada de la UX. Finalmente, implementar funcionalidades en el entorno web para trabajar con datos no estructurados, permitiría tener una herramienta para procesos de *clustering* más holística.

spanish

REFERENCIAS

[1] E. Rosenzweig, *Successful user experience: Strategies and roadmaps*. Morgan Kaufmann, 2015.

[2] J. Kiruthika, S. Khaddaj, D. Greenhill, and J. Francik, "User experience design in web applications," in *2016 IEEE Intl Conference on Computational Science and Engineering (CSE) and IEEE Intl Conference on Embedded and Ubiquitous Computing (EUC) and 15th Intl Symposium on Distributed Computing and Applications for Business Engineering (DCABES)*. IEEE, 2016, pp. 642–646.

[3] C. Reis, S. D. Gribble, and H. M. Levy, "Architectural principles for safe web programs." in *HotNets*, 2007.

[4] M. E. Celebi and K. Aydin, *Unsupervised learning algorithms*. Springer, 2016, vol. 9.

[5] A. Saxena, M. Prasad, A. Gupta, N. Bharill, O. P. Patel, A. Tiwari, M. J. Er, W. Ding, and C.-T. Lin, "A review of clustering techniques and developments," *Neurocomputing*, vol. 267, pp. 664–681, 2017.

[6] Y. Reddy, P. Viswanath, and B. E. Reddy, "Semi-supervised learning: A brief review," *Int. J. Eng. Technol*, vol. 7, no. 1.8, p. 81, 2018.

[7] W. Tang, Y. Yang, L. Zeng, and Y. Zhan, "Size constrained clustering with milp formulation," *IEEE Access*, vol. 8, pp. 1587–1599, 2019.

[8] S. Zhang, H.-S. Wong, and D. Xie, "Semi-supervised clustering with pairwise and size constraints," in *2014 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2014, pp. 2450–2457.

[9] W. Tang, Y. Yang, L. Zeng, and Y. Zhan, "Optimizing mse for clustering with balanced size constraints, symmetry, 11 (2019), 338."

[10] J. Bassil, A. Makhoul, B. Piranda, and J. Bourgeois, "Distributed size-constrained clustering algorithm for modular robot-based programmable matter," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 18, no. 1, pp. 1–21, 2023.

[11] N. Ganganath, C.-T. Cheng, and K. T. Chi, "Data clustering with cluster size constraints using a modified k-means algorithm," in *2014 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*. IEEE, 2014, pp. 158–161.

[12] S. Zhu, D. Wang, and T. Li, "Data clustering with size constraints," *Knowledge-Based Systems*, vol. 23, no. 8, pp. 883–889, 2010.

[13] D. Vallejo Huanga, P. Morillo, and C. Ferri, "Semi-supervised clustering algorithms for grouping scientific articles," *Procedia Computer Science*, vol. 108, pp. 325–334, 2017.

[14] P. L. Burk, "Jammin'on the web-a new client/server architecture for multi-user musical performance." in *ICMC*. Citeseer, 2000.

[15] M. Čihák, "Introduction to applied stress testing," 2007.

[16] D. Draheim, J. Grundy, J. Hosking, C. Lutteroth, and G. Weber, "Realistic load testing of web applications," in *Conference on Software Maintenance and Reengineering (CSMR'06)*. IEEE, 2006, pp. 11–pp.

[17] A. Chakravarty, "Stress testing an ai based web service: A case study," in *2010 Seventh International Conference on Information Technology: New Generations*. IEEE, 2010, pp. 1004–1008.

[18] N. S. Aziz, N. S. Sulaiman, W. N. I. T. M. Hassan, N. L. Zakaria, and A. Yaacob, "A review of website measurement for website usability evaluation," in *Journal of Physics: Conference Series*, vol. 1874, no. 1. IOP Publishing, 2021, p. 012045.

[19] L. Wiener, T. Ekholm, and P. Haller, "Modular responsive web design using element queries," *arXiv preprint arXiv:1511.01223*, 2015.