

**INSTITUTO TECNOLÓGICO DE TOLUCA**

**INFORME FINAL DE RESIDENCIA PROFESIONAL**

**“ANÁLISIS DE DATOS CON EL ALGORITMO DE AGRUPAMIENTO  
K-PROTOTYPES”**

**Ingeniería en Sistemas Computacionales**

**PRESENTA:**

**Fabiola Ramírez Guerrero**

**No. CONTROL:**

**C18280980**

**ASESOR INTERNO:**

**Andrea Olimpia Nájera Ávila**

**ASESOR EXTERNO:**

**Dra. Eréndira Rendón Lara**

**METEPEC, ESTADO DE MÉXICO, OCTUBRE DE 2024**

Agradecimientos

Quisiera aprovechar esta oportunidad para expresar mi más sincero agradecimiento a todas las personas que han estado a mi lado, ofreciéndome su apoyo a lo largo de mi carrera universitaria y en la realización de este proyecto de residencia.

Extiendo mi gratitud a mis profesores por su compromiso, por compartir conmigo su vasto conocimiento y experiencia. Su dedicación ha sido una fuente constante de inspiración y me ha motivado a seguir adelante, permitiéndome culminar esta etapa tan importante en mi vida.

A mi familia, pero en especial les agradezco profundamente por su incondicional apoyo, por acompañarme en los buenos y malos momentos. Su amor, paciencia y comprensión han sido esenciales para llegar a donde estoy hoy. Mamá, papá ustedes me han enseñado que el éxito no es solo la culminación de un proyecto o la obtención de un título, sino el proceso de crecer, aprender y seguir adelante, incluso cuando las cosas no salen como uno espera. Me han mostrado que, aunque el camino sea arduo, con esfuerzo, disciplina y el apoyo de quienes nos aman, todo es posible.

A mi compañero, quiero agradecerte profundamente por estar a mi lado en cada paso de este camino. Tu apoyo incondicional, tus palabras de aliento y tu confianza en mí han sido fundamentales para llegar hasta aquí. No podría haber logrado esto sin tu presencia, y por ello, te dedico este logro con todo mi aprecio y gratitud.

Finalmente, quisiera expresar mi reconocimiento al Instituto Tecnológico de Toluca por ofrecerme la valiosa oportunidad de cursar mis estudios y formarme como profesional. Esta institución no solo me proporcionó conocimientos académicos, sino también una sólida base de principios y habilidades que serán clave en mi desarrollo personal y profesional.

Este logro no habría sido posible sin el apoyo y confianza de todos ellos. Les estaré eternamente agradecido.

Resumen

Este proyecto se centra en la implementación del algoritmo K-Prototypes, un método de agrupamiento diseñado para trabajar con conjuntos de datos que combinan variables numéricas y categóricas. El algoritmo K-Prototypes supera las limitaciones de otros enfoques tradicionales, como el K-Means, al integrar en su cálculo tanto la distancia euclidiana para los datos numéricos como la disimilitud de coincidencia para los categóricos. Este enfoque híbrido lo convierte en una herramienta ideal para análisis de datos heterogéneos.

A lo largo del desarrollo del proyecto, se implementaron los componentes clave del algoritmo, desde la inicialización aleatoria de prototipos hasta la asignación iterativa de objetos a clusters, y se ajustaron las métricas de distancia para garantizar una correcta clasificación de los datos mixtos. Se buscó optimizar el algoritmo en términos de rendimiento, asegurando que pudiera manejar eficientemente grandes volúmenes de datos.

El proceso de agrupamiento permitió generar clusters homogéneos y coherentes, lo que facilita el análisis de patrones complejos en los datos. Los resultados obtenidos validan la eficacia del algoritmo K-Prototypes para abordar la segmentación en bases de datos que contienen tanto atributos numéricos como categóricos, demostrando su versatilidad en diversas áreas de aplicación.

Este proyecto no solo aborda la correcta implementación del algoritmo, sino que también destaca su aplicabilidad y valor en el contexto del análisis de datos. Los logros alcanzados reflejan un avance en la exploración de técnicas modernas para trabajar con datos mixtos, brindando un punto de partida sólido para futuras investigaciones y aplicaciones.

Índice

[1 Introducción 1](#_Toc178547273)

[2 Descripción de la empresa u organización y del puesto o área del trabajo del estudiante. 3](#_Toc178547274)

[3 Problema a resolver 4](#_Toc178547275)

[4 Objetivos 6](#_Toc178547276)

[4.1 General 6](#_Toc178547277)

[4.2 Específicos 6](#_Toc178547278)

[5 Justificación 7](#_Toc178547279)

[6 Marco Teórico (fundamentos teóricos) 8](#_Toc178547280)

[6.1 Minería de datos 8](#_Toc178547281)

[6.1.1 Disciplinas relacionadas con minería de datos 8](#_Toc178547282)

[6.2 Clustering 9](#_Toc178547283)

[6.2.1 Objetivos del Clustering 9](#_Toc178547284)

[6.2.2 Importancia del Clustering en el Análisis de Datos 10](#_Toc178547285)

[6.2.3 Aplicaciones del Clustering 10](#_Toc178547286)

[6.3 Algoritmos de agrupamiento (clustering) 10](#_Toc178547287)

[6.3.1 Clustering jerárquico 11](#_Toc178547288)

[6.3.2 Clustering particional 12](#_Toc178547289)

[6.3.3 Algoritmos de agrupamiento de recorridos simples. 14](#_Toc178547290)

[6.4 Similaridad Y Disimilaridad 14](#_Toc178547291)

[6.5 Algoritmos de Clustering 15](#_Toc178547292)

[6.5.1 El algoritmo DBSCAN 15](#_Toc178547293)

[6.5.2 El algoritmo BIRCH 15](#_Toc178547294)

[6.5.3 El Algoritmo K-Means 16](#_Toc178547295)

[6.5.4 El algoritmo k-modes 17](#_Toc178547296)

[6.5.5 Algoritmo k-Prototypes 18](#_Toc178547297)

[6.6 Diagrama de flujo K-Prototype 23](#_Toc178547298)

[6.7 Prueba de escritorio 24](#_Toc178547299)

[6.8 Investigaciones y proyectos usados con el algoritmo K-Prototypes 35](#_Toc178547300)

[6.8.1 Mejora de K-Prototypes mediante optimización evolutiva 35](#_Toc178547301)

[6.8.2 Aplicaciones en bioinformática 35](#_Toc178547302)

[6.8.3 Optimización del Transporte Público en Singapur 35](#_Toc178547303)

[6.8.4 Segmentación de Clientes en Comercio Electrónico en China 35](#_Toc178547304)

[6.8.5 Análisis de Datos Clínicos en Estados Unidos 36](#_Toc178547305)

[6.9 Herramientas utilizadas 36](#_Toc178547306)

[7 Procedimiento y descripción de las actividades realizadas. 39](#_Toc178547307)

[7.1 Requisitos funcionales 39](#_Toc178547308)

[7.2 Requisitos no funcionales 42](#_Toc178547309)

[7.3 Análisis de la base de datos 44](#_Toc178547310)

[7.4 Diseño y prototipado 44](#_Toc178547311)

[7.4.1 Diagrama de casos de uso 44](#_Toc178547312)

[7.4.2 Diagrama de secuencias 45](#_Toc178547313)

[7.4.3 Diagrama de clases 46](#_Toc178547314)

[7.5 Desarrollo del sistema 46](#_Toc178547315)

[7.6 Etapa de pruebas 65](#_Toc178547316)

[8 Resultados 66](#_Toc178547317)

[8.1 Vista de usuario 66](#_Toc178547318)

[9 Conclusiones de Proyecto 74](#_Toc178547319)

[10 Recomendaciones 75](#_Toc178547320)

[11 Competencias desarrolladas y/o aplicadas 75](#_Toc178547321)

[11.1 Competencias Específicas 76](#_Toc178547322)

[11.2 Competencias Genéricas 76](#_Toc178547323)

[12 Fuentes de información 77](#_Toc178547324)

[13 Anexos 79](#_Toc178547325)

Índice de figuras

[Figura 1 Representación de un algoritmo de agrupamiento o clustering 11](#_Toc178537856)

[Figura 2 Agrupamiento jerárquico 12](#_Toc178537857)

[Figura 3 Clustering particional 14](#_Toc178537858)

[Figura 4 Diagrama de flujo k-prototype 23](#_Toc178537859)

[Figura 5 Estructura de archivos del proyecto 45](#_Toc178537860)

[Figura 6 Carpeta documents y collections 45](#_Toc178537861)

[Figura 7 Visualización de carpetas con iteraciones finales 46](#_Toc178537862)

[Figura 8 Archivos txt con resultados finales 46](#_Toc178537863)

[Figura 9 Estructura de carpetas 46](#_Toc178537864)

[Figura 10 Levantar los datos 47](#_Toc178537865)

[Figura 11 Importar archivo 48](#_Toc178537866)

[Figura 12 Clase inicial maestra 49](#_Toc178537867)

[Figura 13 Base de datos 50](#_Toc178537868)

[Figura 14 Administrador de sesión de la bdd 50](#_Toc178537869)

[Figura 15 Registros de la tabla 50](#_Toc178537870)

[Figura 16 Obtención números aleatorios 51](#_Toc178537871)

[Figura 17 Obtención de la tabla 51](#_Toc178537872)

[Figura 18 Obtención de cabezales de cada tabla 51](#_Toc178537873)

[Figura 19 Proyección de los campos de la tabla 52](#_Toc178537874)

[Figura 20 Discriminación de prototipos 52](#_Toc178537875)

[Figura 21 Lectura de archivos 52](#_Toc178537876)

[Figura 22 Inserción de archivo 53](#_Toc178537877)

[Figura 23 Consulta de la tabla que se usa 53](#_Toc178537878)

[Figura 24 Iteraciones del K-Prototype 54](#_Toc178537879)

[Figura 25 Operaciones del algoritmo K-Prototype 55](#_Toc178537880)

[Figura 26 Checar e iterar 56](#_Toc178537881)

[Figura 27 Nuevos prototipos de cada iteración 56](#_Toc178537882)

[Figura 28 Método para eliminar caracteres especiales 57](#_Toc178537883)

[Figura 29 Exporta las bases de datos de cualquier tipo de archivo 57](#_Toc178537884)

[Figura 30 Documentos necesarios por iteración 59](#_Toc178537885)

[Figura 31 Interfaz gráfica 1 59](#_Toc178537886)

[Figura 32 Interfaz gráfica 2 59](#_Toc178537887)

[Figura 33 Interfaz gráfica 3 60](#_Toc178537888)

[Figura 34 Interfaz gráfica 4 60](#_Toc178537889)

[Figura 35 Vista total de iteraciones 61](#_Toc178537890)

[Figura 36 Vista principal 62](#_Toc178537891)

[Figura 37 Vista inicialización de ventana principal 62](#_Toc178537892)

[Figura 38 Error de la base de datos 63](#_Toc178537893)

[Figura 39 Aviso repetir proceso 63](#_Toc178537894)

[Figura 40 Carpetas de carga de archivos 64](#_Toc178537895)

[Figura 41 Carga de archivos 64](#_Toc178537896)

[Figura 42 Conexión con la base de datos 64](#_Toc178537897)

[Figura 43 Path de la base de datos 65](#_Toc178537898)

[Figura 44 Base de datos cargada a DBeaver 65](#_Toc178537899)

[Figura 45 Archivo cargado en DBeaver 66](#_Toc178537900)

[Figura 46 Ejecución en consola del sistema 66](#_Toc178537901)

[Figura 47 Ruta para entrar al proceso 66](#_Toc178537902)

[Figura 48 Interfaz inicial 66](#_Toc178537903)

[Figura 49 Interfaz visualización número de iteraciones 67](#_Toc178537904)

[Figura 50 Visualización datos de cada prototipo 67](#_Toc178537905)

[Figura 51 Datos agrupamiento k0 68](#_Toc178537906)

[Figura 52 Datos agrupamiento k1 68](#_Toc178537907)

[Figura 53 Datos agrupamiento k2 69](#_Toc178537908)

[Figura 54 Carpeta documents 69](#_Toc178537909)

[Figura 55 Carpeta collections 70](#_Toc178537910)

[Figura 56 Archivos txt guardados en automático 71](#_Toc178537911)

Índice de tablas

[Tabla 1 Categoría de productos 25](#_Toc178529370)

[Tabla 2 Centros iniciales 26](#_Toc178529371)

[Tabla 3 Iteración 1 distancia objeto 1 26](#_Toc178529372)

[Tabla 4 Iteración 1 distancia objeto 2 26](#_Toc178529373)

[Tabla 5 Iteración 1 distancia objeto 3 27](#_Toc178529374)

[Tabla 6 Iteración 1 distancia objeto 4 27](#_Toc178529375)

[Tabla 7 Iteración 1 distancia objeto 5 28](#_Toc178529376)

[Tabla 8 Asignación de objetos 28](#_Toc178529377)

[Tabla 9 Reasignación de objetos, grupo 1 29](#_Toc178529378)

[Tabla 10 Reasignación de objetos, grupo 2 29](#_Toc178529379)

[Tabla 11 Iteración 2 distancia objeto 1 29](#_Toc178529380)

[Tabla 12 Iteración 2 distancia objeto 2 30](#_Toc178529381)

[Tabla 13 Iteración 2 distancia objeto 3 30](#_Toc178529382)

[Tabla 14 Iteración 2 distancia objeto 4 31](#_Toc178529383)

[Tabla 15 Iteración 2 distancia objeto 5 31](#_Toc178529384)

[Tabla 16 Asignación de objetos segunda iteración 32](#_Toc178529385)

[Tabla 17 Reasignación de objetos, grupo 1 32](#_Toc178529386)

[Tabla 18 Reasignación de objetos, grupo 2 33](#_Toc178529387)

[Tabla 19 Iteración 3 distancia objeto 1 33](#_Toc178529388)

[Tabla 20 Iteración 3 distancia objeto 2 33](#_Toc178529389)

[Tabla 21 Iteración 3 distancia objeto 3 34](#_Toc178529390)

[Tabla 22 Iteración 3 distancia objeto 4 34](#_Toc178529391)

[Tabla 23 Iteración 3 distancia objeto 5 35](#_Toc178529392)

[Tabla 24 Asignación de objetos tercera iteración 35](#_Toc178529393)

[Tabla 25 Requisitos funcionales 40](#_Toc178529394)

[Tabla 26 Requisitos no funcionales 43](#_Toc178529395)

# Introducción

El análisis de datos ha evolucionado significativamente en las últimas décadas, con el surgimiento de métodos más avanzados para extraer información valiosa a partir de grandes volúmenes de datos. En muchos casos, los datos no son homogéneos y están compuestos tanto por variables numéricas como categóricas, lo que presenta un reto al momento de aplicar algoritmos tradicionales de análisis, como el K-Means, que solo trabaja con datos numéricos.

Para abordar esta limitación, el algoritmo K-Prototypes surge como una solución eficaz, permitiendo el procesamiento conjunto de variables numéricas y categóricas. Este algoritmo combina las fortalezas de K-Means (para variables numéricas) y K-Modes (para variables categóricas), lo que lo convierte en una herramienta adecuada para la segmentación de datos mixtos. Su aplicación es fundamental en áreas como el marketing, donde es común encontrar información demográfica de clientes combinada con preferencias o comportamientos categóricos.

En este documento se aborda el desarrollo y aplicación del algoritmo K-Prototypes, tomando como base un conjunto de datos reales compuesto por atributos numéricos, como la edad e ingresos de los clientes, y categóricos, como su estado de cliente y categoría de producto favorito. La implementación de este algoritmo permitirá agrupar a los clientes en clusters significativos, con el objetivo de identificar patrones ocultos que ayuden a la toma de decisiones estratégicas.

El estudio realizado tiene como finalidad no solo evaluar la efectividad del K-Prototypes en la segmentación de datos mixtos, sino también determinar su capacidad para mejorar la precisión y consistencia en la formación de clusters comparado con otros métodos de agrupación. Esto resulta particularmente relevante dado el creciente uso de datos heterogéneos en diversas industrias y la necesidad de contar con algoritmos que puedan analizarlos de manera conjunta y eficaz.

En este contexto, se espera que la implementación del algoritmo K-Prototypes proporcione resultados robustos y confiables que sirvan como base para la toma de decisiones basada en datos, superando las limitaciones de los algoritmos que solo manejan un tipo de variable. Con ello, se busca contribuir al desarrollo de técnicas de análisis más precisas y adaptadas a la realidad de los datos mixtos que cada vez son más comunes en el análisis de datos moderno.

# Descripción de la empresa u organización y del puesto o área del trabajo del estudiante.

El Instituto Tecnológico de Toluca pertenece al sistema de la Secretaría de Educación Pública conocido como Tecnológico Nacional de México. Esta red se encarga de formar profesionales en la Ciencia y la Tecnológica, así como impulsarlos a través de la creación de proyectos y su divulgación.

El ITT se ha consolidado como una de las universidades de calidad en el Estado de México con respecto a esta área. Han tratado de cubrir las áreas industriales y de servicios en varias regiones del país, vinculándose a empresas, organizaciones e instituciones. Promueven entre sus catedráticos y estudiantes valores como la justicia, independencia, democracia y solidaridad. Por eso procuran el deporte y la cultura como estrategia para mentes y cuerpos sanos.

Realizan actividades donde vinculan a sus estudiantes a proyectos con comunidades, así como la movilidad a otras ciudades o países. Y también imparten capacitación continua para sus egresados y personas en general que quieran seguirse preparando.

Misión

Ofrecer servicios de Educación Superior Tecnológica con calidad, equidad y pertinencia, orientados a la formación integral del ser humano mediante la oferta de programas educativos acreditados y que promuevan el desarrollo sustentable para la conformación de una sociedad justa y humana.

Visión

Ser una Institución de Educación Superior Tecnológica basada en valores y reconocida por ofrecer un servicio de calidad, mediante el desarrollo sostenido, sustentable y equitativo de su entorno y a través del alto desempeño de su personal.

# Problema a resolver

Actualmente, las organizaciones se enfrentan a desafíos significativos al intentar analizar grandes conjuntos de datos que contienen tanto variables numéricas como categóricas. Estos datos mixtos son comunes en muchos campos, desde el comercio minorista hasta la investigación científica, y la falta de herramientas adecuadas para procesarlos y agruparlos eficazmente representa un obstáculo importante.

Los métodos tradicionales de análisis de datos, como los algoritmos de agrupamiento estándar, suelen ser efectivos solo con un tipo de variable, ya sea numérica o categórica, pero no ambos. Esta limitación impide la identificación de patrones y tendencias clave en los datos mixtos, lo que puede llevar a interpretaciones incompletas y decisiones menos efectivas. La incapacidad de manejar estos datos de manera eficiente resulta en una visión parcial de las relaciones importantes entre las variables numéricas y categóricas, afectando negativamente la toma de decisiones estratégicas.

Además, la integración y comprensión de datos mixtos es crucial para diversas aplicaciones, desde el análisis de comportamiento del cliente hasta la investigación científica y la gestión operativa. Sin herramientas adecuadas, las organizaciones tienen dificultades para integrar datos de diferentes fuentes y obtener una comprensión holística. Esta deficiencia limita la capacidad para desarrollar estrategias efectivas y mejorar procesos operativos.

Existe una necesidad urgente de herramientas que puedan manejar eficientemente datos mixtos, permitiendo la identificación de patrones y tendencias que no son evidentes cuando se analizan tipos de datos por separado. Un sistema que utilice algoritmos avanzados, como el K-Prototypes, puede agrupar datos mixtos en clústeres significativos, mejorando la precisión del análisis y facilitando la toma de decisiones basada en datos.

Sin una solución adecuada, las organizaciones continuarán enfrentando dificultades para extraer valor significativo de sus datos, lo que puede resultar en decisiones subóptimas y limitar la capacidad de innovación. En un entorno empresarial competitivo, la capacidad de analizar y actuar sobre datos de manera efectiva es un diferenciador clave. La falta de herramientas adecuadas puede poner a las organizaciones en desventaja frente a competidores que sí pueden aprovechar completamente sus datos.

# Objetivos

## General

Implementación del algoritmo de agrupamiento K-Prototypes para encontrar la descripción de patrones en un conjunto de datos con datos numéricos y categóricos.

## Específicos

1. Estudio y análisis de algoritmos de agrupamiento (K-Prototypes)
2. Análisis y comprensión de la base de datos que será utilizada
3. Análisis estadístico de la base de datos que se utilizará
4. Diseño del prototipo del Software
5. Implementación en el lenguaje Phyton
6. Realización de pruebas
7. Presentación de resultados

# Justificación

Existen pocos algoritmos de agrupamiento que trabajen con datos de tipo mezclado (Numérico y categóricos) por esta razón, contar con esta implementación del algoritmo de agrupamiento permitirá, realizar estudios de análisis de datos, para cualquier conjunto de datos de este tipo. Muchos conjuntos de datos contienen una combinación de variables numéricas y categóricas, lo que presenta un desafío considerable para los métodos de agrupamiento tradicionales. La falta de herramientas adecuadas para analizar estos datos mixtos puede llevar a interpretaciones incompletas y decisiones menos efectivas.

La implementación del algoritmo de agrupamiento K-Prototypes permitirá analizar conjuntos de datos mixtos de manera más precisa y eficiente. Este proyecto se enfoca en desarrollar un software que utilice dicho algoritmo para identificar patrones y tendencias en los datos, lo que facilitará una comprensión más profunda de la información disponible.

Al permitir un análisis más completo de estos datos, se podrá mejorar la gestión de recursos, el rendimiento del sistema, y adaptar sus estrategias a las necesidades y comportamientos de los usuarios. La integración del algoritmo K-Prototypes es una herramienta accesible que permita a los usuarios realizar análisis complejos sin necesidad de una formación avanzada en análisis de datos.

El desarrollo de este software no solo solucionará un problema técnico significativo, sino que también abrirá nuevas oportunidades para el análisis de datos. Al ofrecer una solución robusta para el análisis de datos mixtos, el proyecto contribuirá a la toma de decisiones más informadas y estratégicas, mejorando así la eficiencia y efectividad operativa.

# Marco Teórico (fundamentos teóricos)

## Minería de datos

Hoy en día, las empresas que gestionan grandes volúmenes de datos buscan formas de analizarlos para identificar oportunidades que les permitan monitorear las ventas y dirigir sus estrategias de marketing. También buscan detectar tendencias que puedan predecir situaciones futuras. En esencia, el objetivo es identificar modelos o patrones que den sentido a la vasta cantidad de datos disponibles. Esta tarea recae en la minería de datos (MD), que se enfoca en extraer información valiosa o conocimiento a través del análisis detallado de grandes conjuntos de datos, utilizando técnicas y metodologías (algoritmos) diseñados para la clasificación, regresión, agrupamiento y detección de anomalías, entre otras.

El proceso de minería de datos implica ajustar modelos o identificar patrones a partir de datos observados. Este ajuste es generalmente de naturaleza estadística, permitiendo cierto nivel de ruido o error dentro del modelo. Los algoritmos de minería de datos se encargan principalmente de tareas de predicción (para datos desconocidos) y descripción (de patrones). Los objetivos de la predicción y la descripción se logran mediante las tareas fundamentales de la minería de datos: clasificación, regresión, agrupamiento, resumir información, dependencia del modelo, y detección de cambios y desviaciones. La mayoría de los métodos de minería de datos están basados en conceptos de aprendizaje automático, reconocimiento de patrones y estadística. El amplio conjunto de algoritmos disponibles para resolver estos problemas puede resultar abrumador tanto para los analistas de datos experimentados como para los principiantes. Los desarrollos actuales en minería de datos se centran en encontrar métodos que clasifiquen eficientemente grandes bases de datos, con datos de diversas naturalezas o combinaciones de estos.

.

### Disciplinas relacionadas con minería de datos

Por definición, el KDD (Descubrimiento de Conocimiento en Bases de Datos) se considera un campo interdisciplinario que reúne a investigadores de diversas áreas. El término "minería de datos" ha sido comúnmente utilizado por estadísticos, analistas de datos y la comunidad de Sistemas de Información para la Gestión (MIS), mientras que "KDD" ha sido más empleado en inteligencia artificial y en investigaciones de aprendizaje automático.

Estadística

La estadística ofrece una valiosa asistencia en el análisis de datos, pero a menudo no es suficiente por sí sola y presenta ciertos inconvenientes, como la inadecuación para datos nominales, la dificultad en la interpretación de resultados y la necesidad de que el usuario decida cómo y dónde analizar los datos. No obstante, desempeña un papel crucial en algunos pasos del proceso de KDD, especialmente en la selección y muestreo de datos, en la minería de datos y en la evaluación del conocimiento obtenido.

Inteligencia Artificial y Aprendizaje Automático

Estas disciplinas aportan algoritmos y métodos esenciales para el aprendizaje automático, permitiendo que los sistemas identifiquen patrones y hagan predicciones a partir de los datos. Los enfoques de aprendizaje supervisado y no supervisado son fundamentales para desarrollar modelos predictivos y de agrupamiento.

Bases de Datos y Sistemas de Información

El almacenamiento y recuperación eficiente de datos son vitales en la minería de datos. Las bases de datos relacionales y NoSQL, junto con los sistemas de gestión de bases de datos (DBMS), facilitan la organización y el acceso a grandes conjuntos de datos, apoyando las operaciones de minería.

Matemáticas y Algoritmos

La teoría matemática sustenta muchas técnicas de minería de datos. Los algoritmos de optimización, la teoría de grafos y las matemáticas discretas son esenciales para desarrollar métodos eficientes y efectivos para el análisis de datos.

Visualización de Datos

La capacidad de visualizar datos es crucial para interpretar y comunicar los resultados de la minería de datos. Herramientas y técnicas de visualización ayudan a presentar patrones complejos de manera comprensible y a facilitar la toma de decisiones basadas en datos.

Ciencias de la Computación

Las ciencias de la computación proporcionan los fundamentos técnicos y teóricos necesarios para el desarrollo de software y herramientas de minería de datos. Esto incluye desde el diseño de algoritmos hasta la implementación de sistemas escalables y eficientes.

## Clustering

Clustering es una técnica de aprendizaje no supervisado que se utiliza para agrupar un conjunto de objetos en subgrupos o clusters, de manera que los objetos en un mismo cluster sean más similares entre sí que con los objetos de otros clusters. La similitud entre objetos se define a menudo mediante una medida de distancia, como la distancia Euclidiana, aunque puede variar según el tipo de datos y el algoritmo utilizado.

El Clustering es un enfoque o método popular usado para segmentar grandes conjuntos de datos heterogéneos, dentro de pequeños subconjuntos homogéneos que puedan ser fácilmente manejados, analizados y modelados por separado. Los métodos de agrupamiento reparten un conjunto de objetos dentro de grupos, de tal forma que los objetos en el mismo grupo son más similares mutuamente que objetos en diferentes grupos conforme a algún criterio definido. El Clustering ha sido estudiado en diversas áreas, cada una tiene sus métodos para organizar los datos, en las áreas de análisis de datos y estadística se requiere a los datos en forma de números (cuantitativos). En aprendizaje de máquina, en forma de conceptos y simbolos (cualitativos). En minería de datos frecuentemente se trata con todos los tipos de datos o con la mezcla de estos: cuantitativos (numéricos) y cualitativos (categóricos o simplificación de simbólicos).

### Objetivos del Clustering

Descubrimiento de estructuras: Identificar patrones ocultos en los datos sin etiquetas predefinidas.

Reducción de dimensionalidad: Simplificar grandes conjuntos de datos agrupándolos en clusters representativos.

Detección de anomalías: Identificar puntos de datos que no pertenecen a ningún cluster, lo que puede ser útil para detectar fraudes o fallos.

Segmentación de mercados: Dividir un mercado en segmentos distintos para identificar grupos de clientes con comportamientos similares.

### Importancia del Clustering en el Análisis de Datos

El clustering es una herramienta fundamental en el análisis de datos y tiene aplicaciones en diversas disciplinas. En biología, por ejemplo, se usa para clasificar especies similares basadas en características genéticas. En el comercio electrónico, ayuda a segmentar a los clientes en grupos basados en su comportamiento de compra, permitiendo la personalización de campañas de marketing. En seguridad informática, se emplea para detectar patrones anómalos que podrían indicar ataques cibernéticos.

### Aplicaciones del Clustering

Biología y Medicina: Agrupamiento de genes con funciones similares, clasificación de enfermedades.

Marketing y Ventas: Segmentación de clientes, análisis de preferencias de consumidores.

Redes Sociales: Detección de comunidades en redes sociales, análisis de opiniones y sentimientos.

Seguridad y Fraude: Detección de transacciones fraudulentas, identificación de patrones inusuales en datos de seguridad.

Procesamiento del Lenguaje Natural: Agrupamiento de documentos o términos, análisis de sentimientos.

El clustering no solo organiza los datos en grupos manejables, sino que también facilita el descubrimiento de relaciones complejas y patrones en los datos que no son fácilmente visibles. Esto permite una comprensión más profunda y una toma de decisiones más informada en una variedad de campos.

## Algoritmos de agrupamiento (clustering)

Los algoritmos de agrupamiento reciben como entrada al conjunto de datos o la misma base de datos y sin supervisión de ningún tipo y de forma autónoma, agrupan los datos dentro de k grupos o clusters. Clustering es útil en ciertas aplicaciones de biología, medicina, sociología y otras donde no se encuentran bien definidas las clases. La figura [1] muestra la representación de un algoritmo de agrupación, la entrada del algoritmo se representa por el conjunto de datos u objetos y dependiendo del algoritmo de agrupamiento se solicita o no el parámetro k; la salida, es una etiqueta para cada uno de los elementos del conjunto de datos, la cual indica que dicho dato pertenece a cierta clase, grupo o cluster.

Tabla

Descripción generada automáticamente con confianza baja

Figura 1 Representación de un algoritmo de agrupamiento o clustering

Los algoritmos de **agrupamiento conceptual** desarrollados en aprendizaje de máquina agrupan datos con valores categóricos y también producen descripciones conceptuales de los clusters. La descripción conceptual de los grupos es importante en minería de datos, porque ayuda a la interpretación de los resultados del agrupamiento.

Los algoritmos de **agrupamiento estadístico** cuentan con algoritmos basados en la búsqueda de objetos que tengan conceptos iguales o similares y su eficiencia depende de buenas estrategias de búsqueda. Para los problemas de minería de datos que frecuentemente involucran muchos conceptos y espacios muy grandes de objetos, los métodos basados en conceptos pueden ser de potencial desventaja.

### Clustering jerárquico

Organiza los datos en una secuencia anidada de grupos. Una de sus características es el impacto visual del dendograma que habilita al analista de datos a visualizar los objetos mezclados en grupos o divisiones sucesivas de niveles de proximidad, los métodos de agrupamiento generalmente requieren solo el uso de la matriz de proximidad, estas técnicas son populares en las ciencias biológicas, sociales y de comportamiento por que se necesita construir taxonomías. Los algoritmos de clustering jerárquico tienen como objetivo unir grupos (cada grupo de objetos es manejado como si fuera un sólo objeto) para formar uno nuevo, tendiendo hacia un nivel de generalización mayor (métodos aglomerativos) o bien separando grupos formando nuevos sub-agrupamientos de ellos (métodos divisivos), tendiendo hacia un nivel mayor de particularización. Los algoritmos de clustering jerárquico descomponen una base de datos, en varios niveles de clustering. La descomposición jerárquica es representada por un dendograma o árbol, que divide iterativamente las bases de datos en subconjuntos más pequeños hasta que cada subconjunto consta de un objeto. En tal jerarquía, cada nodo del árbol representa un cluster de las bases de datos.

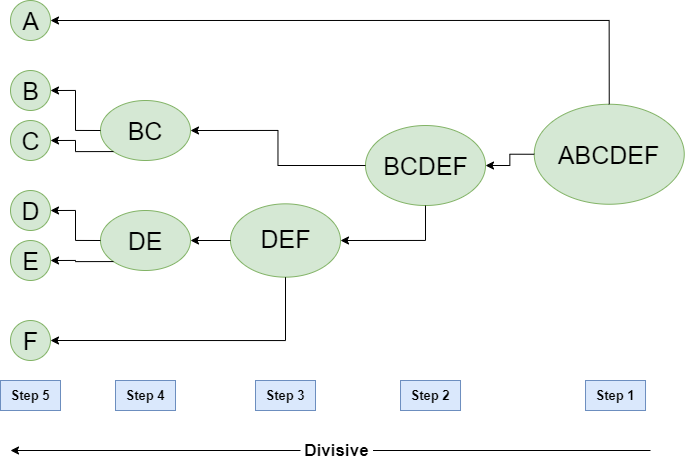


Figura 2 Agrupamiento jerárquico

### Clustering particional

Por su parte el clustering particional genera una simple partición de los datos, en un intento de recuperar los grupos naturales presentes en éstos, para ello cuentan con los datos en forma de matriz de patrones, donde las características han sido medidas en la escala ratio. Las técnicas de partición son usadas para aplicaciones de ingeniería y son apropiadas para una eficiente representación y comprensión de grandes bases de datos. El problema de agrupamiento puede formalizarse de la siguiente manera: Dados patrones en un espacio métrico de d-dimensiones, determinar una partición de los patrones en k cluster o grupos, tal que los patrones en un grupo sean más similares entre si que con otros patrones de diferentes grupos. El valor de k puede o no ser especificado.

Un criterio de agrupamiento como el error cuadrático puede adoptarse, el criterio puede clasificarse como global o local. El criterio global, representa cada grupo por un prototipo y asigna los patrones al grupo de acuerdo con el prototipo más similar. El criterio local, forma grupos utilizando una estructura local en los datos. Por ejemplo, los grupos pueden formarse por la identificación de las regiones de alta densidad en el espacio de patrones o por la asignación de un patrón y los i-ésimos k vecinos más cercanos del mismo grupo. Cada grupo es representado por su centro de gravedad y los objetos restantes son asignados al grupo cuyo objeto representativo sea más cercano.

Estos algoritmos inician con una partición inicial de las bases de datos y entonces usan una estrategia de control iterativa para mover los objetos de un grupo a otro, tratando de optimizar la calidad de los clusters durante el agrupamiento. Pero estos algoritmos presentan las siguientes limitaciones: Pueden aplicarse en problemas en los cuales existen objetos descritos por atributos numéricos exclusivamente, no manejan ausencia de información y muchas veces es necesario especificar el número de grupos que formarán el agrupamiento. Estos algoritmos de partición consideran el agrupamiento como un problema de optimización y pueden sufrir de mínimos locales.

Los algoritmos de partición construyen separan una base de datos en conjuntos de k grupos, donde k puede o no ser un parámetro de entrada. Cada grupo puede ser representado por un centro de gravedad k-means, o por uno de los objetos del grupo localizado cerca de su centro k-medoide, el resto de los objetos son asignados al grupo más cercano. La figura de todos los grupos encontrados por un algoritmo de partición es convexa como en la figura 2.

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

Figura 3 Clustering particional

### Algoritmos de agrupamiento de recorridos simples.

Estos algoritmos agrupan los objetos cercanos del conjunto de objetos utilizando como base una condición local de agrupamiento, recorriendo una sola vez todo el conjunto de objetos. Dentro de esta familia cabe destacar los algoritmos BIRCH y DBSCAN los cuales agrupan conjuntos de datos utilizando criterios locales.

## Similaridad Y Disimilaridad

Una vez que se identifican los tipos de características, el siguiente desafío consiste en determinar qué tan cerca o lejos se encuentran los objetos entre sí. Para abordar este problema, se emplean medidas de similitud y disimilitud. En términos generales, cuando dos objetos son muy similares entre ellos, el valor de la similitud es alto, mientras que la disimilitud es baja. La disimilitud a menudo se mide a través de una distancia, y existen diversas medidas de distancia que dependen del tipo de característica que se esté analizando. Por ejemplo, si las características son categóricas, no se pueden utilizar medidas de distancia que operen en un espacio métrico.

En la literatura, es común encontrar el término "métrica" para referirse a estas medidas de distancia. En este trabajo, nos referiremos a ellas simplemente como "medidas de distancia". Así, una distancia 𝑑(𝑥,𝑦) entre dos objetos 𝑥 y 𝑦 es una función que cumple con las siguientes propiedades propiedades

* 𝑑(𝑥,𝑦) no negativo
* 𝑑(𝑥,𝑦) identidad Si x=y
* 𝑑(𝑥,𝑦) 𝑑(𝑥,𝑦) simetría
* 𝑑(𝑥,𝑦) 𝑑(𝑥,z) + 𝑑(z,y) desigualdad triangular

## Algoritmos de Clustering

### El algoritmo DBSCAN

Este algoritmo trabaja sobre una noción basada en la densidad de los grupos y está diseñado para descubrir grupos de formas arbitrarias e irregulares en bases de datos espaciales con ruido, para ello utiliza estructuras de índices como árboles R\*-Tree y sus variantes. La idea es que cada rama del árbol no contenga más puntos de los preestablecidos en el parámetro "MinPts". Una de las limitaciones que presenta este algoritmo es el manejo de datos espaciales exclusivamente. El agrupamiento generado por el algoritmo puede variar si se cambia el orden de entrada de los objetos del conjunto de datos.

* Principio: Agrupa puntos que están densamente conectados.
* Ventajas: Detecta clusters de formas arbitrarias y maneja ruido (outliers).
* Limitaciones: Depende del parámetro de densidad y puede variar con el orden de los datos.

### El algoritmo BIRCH

Este algoritmo utiliza una estructura de árbol llamada CF-tree (cluster features tree), el cual almacena en sus nodos descripciones compactas de subgrupos (CF's), donde un CF se define como: número de objetos, suma lineal y suma de los cuadrados de todos los objetos en ese grupo. La metodología que utiliza este algoritmo es la de generar un árbol de CF's con el conjunto de datos, para obtener las descripciones de los subgrupos y aplicar posteriormente un algoritmo de clasificación no supervisada para agrupar estos subgrupos. Asi, BIRCH transforma el problema de clasificar el conjunto de datos original en clasificar subgrupos de este conjunto de datos. BIRCH fue diseñado para agrupar grandes bases de datos con atributos numéricos, resumiendo la base de datos original hasta ajustarla al tamaño de la memoria principal disponible en el sistema en que se ejecuta, asi minimiza los costos de lectura y escritura de los datos. Las limitaciones que presenta este algoritmo son las siguientes: puede aplicarse en problemas donde se manejan datos numéricos exclusivamente, y no todas las funciones de distancia pueden aplicarse al utilizar este algoritmo, ya que la información manejada por los CF's permite manipular solamente una familia de medidas que puedan expresarse en términos del centroide, radio y diámetro, dependiendo del orden de entrada de los datos.

* Principio: Utiliza un árbol CF (Cluster Features) para agrupar datos.
* Ventajas: Eficiente en memoria y adecuado para grandes bases de datos numéricas.
* Limitaciones: Solo funciona con datos numéricos y tiene restricciones en las medidas de distancia.

### El Algoritmo K-Means

El algoritmo k-Means fue propuesto hace poco más de tres décadas y es uno de los algoritmos de agrupamiento más usados en una amplia variedad de áreas. k-Means recibe como parámetros el número de agrupamientos a formar y se encuentra definido sobre datos continuos, es decir, únicamente permite trabajar con objetos descritos por medio de un conjunto de atributos numéricos. Este algoritmo calcula iterativamente los centros de los agrupamientos mientras que al mismo tiempo minimiza una función objetivo. k-Means usa la distancia Euclidiana para comparar objetos y promedios para calcular los centros de los agrupamientos, lo que no le permite trabajar con atributos no numéricos.

1. Inicialización: Selecciona k centros de clúster iniciales aleatoriamente del conjunto de datos.
2. Asignación de Clústeres: Asigna cada punto de datos al centro de clúster más cercano usando la distancia euclidiana.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

1. Recalcular Centros: Calcula el nuevo centroide de cada clúster como el promedio de todos los puntos asignados al clúster.

Un dibujo de una persona

Descripción generada automáticamente con confianza media

1. Iterar: Repite los pasos de asignación y re-calculación hasta que los centros no cambien significativamente entre iteraciones.
2. Convergencia: El algoritmo termina cuando los centros de clúster se estabilizan.

### El algoritmo k-modes

Permite extender k-means para ser utilizado con variables categóricas, a partir del cálculo de una medida de disimilitud que permita comparar observaciones categóricas, y la utilización de modas en lugar de medias para calcular los clusters. Por motivos de eficiencia computacional, el algoritmo k-modes sigue estos pasos para un número prestablecido de k clusters:

1. Selecciona k modas, una para cada cluster.
2. Asigna cada observación al cluster con la moda más cercana según la medida de disimilitud.
3. Actualiza la moda de cada cluster después de cada asignación.
4. Recalcula la medida de disimilitud después de que todas las observaciones han sido asignadas, y procede a reubicar aquellas más cercanas a otro cluster que al propio.
5. Recalcula las modas para los clusters afectados.
6. Repite los dos últimos pasos hasta que ninguna observación cambie de cluster después de un ciclo completo para todo el conjunto de datos.

Las principales características del algoritmo K-Modes son:

* Utiliza una medida de disimilitud para comparar objetos categóricos.
* Reemplaza el cálculo de promedios por el de modas.
* Usa un método basado en frecuencias para actualizar las modas de los grupos.

El algoritmo K-Modes sigue una estructura similar al K-Means, pero está adaptado para manejar datos categóricos, lo que lo hace adecuado para aplicaciones donde los datos no son numéricos. La integración de K-Means y K-Modes en el algoritmo K-Prototypes permite trabajar con conjuntos de datos mixtos, combinando la efectividad de ambos métodos.

* Principio: Extiende K-Means para datos categóricos usando una medida de disimilitud basada en frecuencias.
* Ventajas: Agrupa datos categóricos de manera eficiente.
* Limitaciones: Solo trabaja con datos categóricos.

### Algoritmo k-Prototypes

En 1997, Joshua Zhexue Huang presentó el algoritmo K-Prototypes, diseñado para el agrupamiento de datos mixtos.

Uno de los principales desafíos en la minería de datos es cómo particionar grandes conjuntos de datos de manera eficiente en grupos homogéneos, lo que constituye el objetivo central de la tarea descriptiva conocida como clustering o agrupamiento.

Existen métodos de agrupamiento que destacan por su capacidad para manejar grandes volúmenes de datos, pero su aplicación se limita a conjuntos de datos que contienen únicamente atributos numéricos. Sin embargo, muchos conjuntos de datos también incluyen atributos tanto numéricos como categóricos, lo que deja a los métodos tradicionales fuera de consideración para este tipo de datos.

El algoritmo K-Prototypes se basa en el modelo del K-Means, pero supera la limitación de solo trabajar con datos numéricos, ya que está diseñado específicamente para datos mixtos (numéricos y categóricos). Se introduce el concepto de centroide para representar el prototipo de un grupo con atributos mixtos; para los atributos categóricos se utiliza una medida de similitud, mientras que para los atributos numéricos se aplica una ecuación específica para calcular la distancia entre los objetos y los centroides de los grupos. Ambos componentes se combinan para determinar el centroide que representa el prototipo del grupo con atributos mixtos.

Este algoritmo está optimizado para trabajar con grandes conjuntos de datos mixtos, y su principal objetivo es reducir el costo computacional asociado con el agrupamiento de este tipo de datos.

El algoritmo k-Prototypes es un algoritmo de agrupamiento restringido que permite agrupar grandes conjuntos de datos mixtos. Constituye una integración de los algoritmos k-Modes y k-Means. Este algoritmo se basa en el paradigma del algoritmo k-means, preservando su eficiencia, pero eliminando la limitación de trabajar con únicamente datos de tipo numérico.

El algoritmo k-Modes fue la primera extensión del algoritmo k-Means orientada al agrupamiento de datos categóricos. Sigue la misma idea que el k-Means, pero con la principal diferencia en la medida de similitud utilizada para comparar objetos. Sus características principales incluyen el uso de una medida de disimilaridad para comparar objetos, el reemplazo del uso de promedios por el de modas, y un método basado en frecuencias para actualizar las modas. Este algoritmo fue diseñado exclusivamente para agrupar grandes conjuntos de datos categóricos.

El algoritmo k-Prototypes integra tanto al k-Means como al k-Modes para eliminar la limitación de trabajar únicamente con un solo tipo de datos. Asume que la medida de disimilaridad entre atributos numéricos se define por el cuadrado de la distancia Euclidiana y la medida de disimilaridad entre atributos categóricos se define por el número de incoincidencias de categorías entre objetos.

**Funciones de distancia para K-Prototype**

* **Distancia para atributos numéricos o distancia Euclidiana ():**

Sean x, y objetos con atributos numéricos (x1, x2 ,...,xn ) y (y1, y2,...,yn) respectivamente, la distancia Euclidiana se calcula con la la siguiente ecuación:

* **Distancia para atributos categóricos ():**

Para datos categóricos, sean b y c, objetos con atributos categóricos (b1, b2, ,...,bn ) y (c1 ,c2 ,..,cn ) respectivamente, la distancia se calcula con la ecuación:

Donde y es un peso para los atributos categóricos en el prototipo j, en este caso es igual a uno, y...

* **Distancia para atributos mezclados ():**

Sean b y c. dos objetos (b1, b2,...,bn) y (c1 ,c2 ,..,cn ), donde los atributos tienen valores tanto numéricos como categóricos. Para el algoritmo K-Prototypes la distancia que existe entre los objetos b y c, se calcula como la suma de la , más se y denota con la ecuación:

=

**Funcionamiento**

Integración de K-Means y K-Modes:

* K-Means: Utiliza la distancia euclidiana para medir la similitud entre objetos numéricos.
* K-Modes: Utiliza la disimilitud de coincidencia para datos categóricos, basándose en la frecuencia de las categorías.

**Parámetro 𝛾:**

* Controla el balance entre atributos numéricos y categóricos.
* Un valor bajo de 𝛾 da más peso a los atributos numéricos, mientras que un valor alto prioriza los categóricos.

​**Proceso de Clustering:**

Paso 1: Elegir k centros iniciales.

Paso 2: Se eligen k-prototipos aleatoriamente.

Paso 3: Para cada objeto de la base de datos se calcula la distancia (D...) existente entre el objeto y los k-prototipos.

Paso 4: El objeto es asignado al prototipo que presente la menor distancia. En caso de empate seleccione un prototipo al azar, en este caso se toma el primero. (El paso 3 y 4 se efectúa para cada uno de los objetos).

Paso 5: Cuando todos los objetos son asignados a un prototipo los k-prototipos son recalculados.

Paso 6: Los nuevos prototipos son asignados calculando un promedio entre los objetos asignados a ese prototipo

Paso 7: Se recalcula la distancia entre los objetos y los nuevos prototipos, si el objeto presenta una menor distancia entonces el objeto se mueve del prototipo asignado originalmente al prototipo que presente la menor distancia

Paso 8: Se repiten los pasos del 5 al 8 si algún objeto cambió de prototipo en caso contrario terminar. Si después de un número determinado de interacciones el algoritmo no converge termine el algoritmo con las asignaciones actuales. (El paso 7 y 8 se efectúa para cada uno de los objetos)

Paso 9. Fin del algoritmo

Para la prueba de escritorio de este algoritmo el fin del algoritmo será cuando se logre la estabilización de grupos, es decir, cuando no existan movimientos de objetos entre grupos (movimientas=0

**Aplicaciones**

El K-Prototypes es útil en áreas donde los conjuntos de datos contienen tanto atributos numéricos como categóricos, como en estudios de mercado, análisis de clientes, y ciencias sociales. Permite identificar patrones y agrupamientos en datos heterogéneos, ofreciendo una visión más completa que los algoritmos que solo manejan un tipo de dato.

## Diagrama de flujo K-Prototype

Diagrama

Descripción generada automáticamente

Figura 4 Diagrama de flujo k-prototype

## Prueba de escritorio

Tenemos un conjunto de datos con información sobre clientes de una tienda en línea. Cada cliente tiene 4 atributos:

1. Edad (numérico)
2. Ingresos anuales (numérico)
3. Categoría de producto favorito (categórico: Electrónica, Ropa, Comida, Libros)
4. Estado del cliente: (nuevo, regular, fiel)

Tabla 1 Categoría de productos

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Cliente | Edad | Ingresos Anuales ($) | Categoría de Producto Favorito | Estado del cliente |
| 1 | 25 | 40,000 | Electrónica | nuevo |
| 2 | 34 | 50,000 | Ropa | regular |
| 3 | 22 | 30,000 | Comida | fiel |
| 4 | 45 | 70,000 | Libros | nuevo |
| 5 | 23 | 35,000 | Electrónica | fiel |

Aplicación del Algoritmo K-Prototypes

Para datos mixtos (numéricos y categóricos), la disimilitud total entre un cliente 𝑖 y un prototipo 𝑦 se calcula con:

=

1. Inicialización: Seleccionamos aleatoriamente k centros iniciales, 𝑘=2
2. Se eligen k-prototipos aleatoriamente, objetos 1 y 3

k=2 prototipos iniciales.

Tabla 2 Centros iniciales

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cliente | Edad | Ingresos Anuales ($) | Categoría de Producto Favorito | Estado del cliente | Prototipo |
| 1 | 25 | 40,000 | Electrónica | nuevo | K1 |
| 3 | 22 | 30,000 | Comida | fiel | K2 |

1. Para cada objeto de la bdd se calcula la distancia () existente entre el objeto y los k-prototipos

Tabla 3 Iteración 1 distancia objeto 1

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 1** con el prototipo **K1** | | Distancia del **objeto 1** con el prototipo **K2** | |
| Datos numéricos: =((25-25)^2-(40000-40000)^2)^(1/2)=0 | Datos categóricos: (Electrónica, Electrónica)=0 + (nuevo, nuevo)=0 Total=0 | Datos numéricos: =((22-25)^2-(40000-30000)^2)^(1/2)=10000.00045 | Datos categóricos: (Electrónica, comida)=1 + (nuevo, fiel)=1 Total=2 |
| Distancia total: 0 | | Distancia Total: 10002.00045 | |

1. Al objeto 1 se le asigna el valor de **K1**
2. Para cada objeto de la bdd se calcula la distancia () existente entre el objeto y los k-prototipos

Tabla 4 Iteración 1 distancia objeto 2

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 2** con el prototipo **K1** | | Distancia del **objeto 2** con el prototipo **K2** | |
| Datos numéricos: =((34-25)^2-(40000-40000)^2)^(1/2)= 10000.00405 | Datos categóricos: (ropa, electrónica)= 1 + (regular, nuevo)= 1  Total= 2 | Datos numéricos: =((34-22)^2-(50000-30000)^2)^(1/2)=20000.0036 | Datos categóricos: (ropa, comida)= 1 + (regular, fiel)=1 Total=2 |
| Distancia total: 10002.00405 | | Distancia Total: 20002.0036 | |

1. Al objeto 2 se le asigna el valor de **K1**
2. Para cada objeto de la bdd se calcula la distancia () existente entre el objeto y los k-prototipos

Tabla 5 Iteración 1 distancia objeto 3

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 3** con el prototipo **K1** | | Distancia del **objeto 3** con el prototipo **K2** | |
| Datos numéricos: =((22-25)^2-(30000-40000)^2)^(1/2)= 10000.00045 | Datos categóricos: (comida, electrónica)= 1+ (fiel, nuevo)= 1  Total=2 | Datos numéricos: =((22-22)^2-(30000-30000)^2)^(1/2)=0 | Datos categóricos: (comida, comida)=0 + (fiel, fiel)=0  Total=0 |
| Distancia total: 10002.00045 | | Distancia Total: 0 | |

1. Al objeto 3 se le asigna el valor de **K2**
2. Para cada objeto de la bdd se calcula la distancia () existente entre el objeto y los k-prototipos

Tabla 6 Iteración 1 distancia objeto 4

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 4** con el prototipo **K1** | | Distancia del **objeto 4** con el prototipo **K2** | |
| Datos numéricos: =((45-25)^2-(70000-40000)^2)^(1/2)= 30000.00667 | Datos categóricos: (libros, electrónica)= 1+ (nuevo, nuevo)= 0  Total=1 | Datos numéricos: =((45-22)^2-(70000-30000)^2)^(1/2)=40000.00661 | Datos categóricos: (libros, comida)=1 + (nuevo, fiel)=1  Total=2 |
| Distancia total: 30001.00667 | | Distancia Total: 40002.00661 | |

1. Al objeto 4 se le asigna el valor de **K1**
2. Para cada objeto de la bdd se calcula la distancia () existente entre el objeto y los k-prototipos

Tabla 7 Iteración 1 distancia objeto 5

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 5** con el prototipo **K1** | | Distancia del **objeto 5** con el prototipo **K2** | |
| Datos numéricos: =((23-25)^2-(35000-40000)^2)^(1/2)= 5000.0004 | Datos categóricos: (electrónica, electrónica)=0 + (fiel, nuevo)=1  Total=1 | Datos numéricos: =((23-22)^2-(35000-30000)^2)^(1/2)=5000.0001 | Datos categóricos: (electrónica, comida)=1 + (fiel, fiel)=0  Total=1 |
| Distancia total: 5001.0004 | | Distancia Total: 5001.0001 | |

1. Al objeto 5 se le asigna el valor de **K2**
2. Todos los objetos han sido asignados en un prototipo como se aprecia en la tabla 8

Tabla 8 Asignación de objetos

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cliente | Edad | Ingresos Anuales ($) | Categoría de Producto Favorito | Estado del cliente | Prototipo |
| 1 | 25 | 40,000 | Electrónica | nuevo | K1 |
| 2 | 34 | 50,000 | Ropa | regular | K1 |
| 3 | 22 | 30,000 | Comida | fiel | K2 |
| 4 | 45 | 70,000 | Libros | nuevo | K1 |
| 5 | 23 | 35,000 | Electrónica | fiel | K2 |

1. Se asignan los nuevos prototipos, para cada prototipo se calcula el promedio de los atributos numéricos y a través de la medida de similaridad y la moda para los atributos categóricos como se muestra en las tablas 9 y 10

Tabla 9 Reasignación de objetos, grupo 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cliente | Edad | Ingresos Anuales ($) | Categoría de Producto Favorito | Estado del cliente | Prototipo |
| 1 | 25 | 40,000 | Electrónica | nuevo | K1 |
| 2 | 34 | 50,000 | Ropa | regular | K1 |
| 4 | 45 | 70,000 | Libros | nuevo | K1 |
| Promedio | 34.67 | 53333.33 | Ropa | nuevo | K1 |

Tabla 10 Reasignación de objetos, grupo 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cliente | Edad | Ingresos Anuales ($) | Categoría de Producto Favorito | Estado del cliente | Prototipo |
| 3 | 22 | 30,000 | Comida | fiel | K2 |
| 5 | 23 | 35,000 | Electrónica | fiel | K2 |
| Promedio | 22.5 | 32500 | Comida | fiel | K2 |

1. Se recalcula la distancia entre los objetos y los nuevos prototipos, si el objeto presenta una menor distancia el objeto se mueve del prototipo asignado originalmente al prototipo que presente la menor distancia

Tabla 11 Iteración 2 distancia objeto 1

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 1** con el prototipo **K1** | | Distancia del **objeto 1** con el prototipo **K2** | |
| Datos numéricos: =((25-34.67)^2-(40000-53333.33)^2)^(1/2)= 13333.33351 | Datos categóricos: (electrónica, ropa)= 1+ (nuevo, nuevo)= 0  Total=1 | Datos numéricos: =((25-22.5)^2-(40000-32500)^2)^(1/2)=7500.000417 | Datos categóricos: (electrónica, comida)=1 + (nuevo, fiel)=1  Total=2 |
| Distancia total: 13334.33351 | | Distancia Total: 7502.000417 | |

1. Al objeto 1 se le asigna el valor de **K1**
2. Se recalcula la distancia entre los objetos y los nuevos prototipos, si el objeto presenta una menor distancia el objeto se mueve del prototipo asignado originalmente al prototipo que presente la menor distancia

Tabla 12 Iteración 2 distancia objeto 2

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 2** con el prototipo **K1** | | Distancia del **objeto 2** con el prototipo **K2** | |
| Datos numéricos: =((34-34.67)^2-(50000-53333.33)^2)^(1/2)= 3333.330067 | Datos categóricos: (ropa, ropa)= 0+ (regular, nuevo)= 1  Total=1 | Datos numéricos: =((34-22.5)^2-(50000-32500)^2)^(1/2)=17500.00378 | Datos categóricos: (ropa, comida)=1 + (regular, fiel)=1  Total=2 |
| Distancia total: 3334.330067 | | Distancia Total: 17502.00378 | |

1. Al objeto 2 se le asigna el valor de **K1**
2. Se recalcula la distancia entre los objetos y los nuevos prototipos, si el objeto presenta una menor distancia el objeto se mueve del prototipo asignado originalmente al prototipo que presente la menor distancia

Tabla 13 Iteración 2 distancia objeto 3

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 3** con el prototipo **K1** | | Distancia del **objeto 3** con el prototipo **K2** | |
| Datos numéricos: =((22-34.67)^2-(53333.33-30000)^2)^(1/2)= 23333.33344 | Datos categóricos: (comidas, ropa)= 1+ (fiel, nuevo)= 2  Total=2 | Datos numéricos: =((22-22.5)^2-(30000-32500)^2)^(1/2)=2500.00005 | Datos categóricos: (comida, comida)=0 + (nuevo, fiel)=1  Total=1 |
| Distancia total: 23335.33344 | | Distancia Total: 2501.00005 | |

1. Al objeto 3 se le asigna el valor de **K2**
2. Se recalcula la distancia entre los objetos y los nuevos prototipos, si el objeto presenta una menor distancia el objeto se mueve del prototipo asignado originalmente al prototipo que presente la menor distancia

Tabla 14 Iteración 2 distancia objeto 4

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 4** con el prototipo **K1** | | Distancia del **objeto 4** con el prototipo **K2** | |
| Datos numéricos: =((45-34.67)^2-(70000-53333.33)^2)^(1/2)= 16666.6732 | Datos categóricos: (libros, ropa)= 1+ (nuevo, nuevo)= 0  Total=1 | Datos numéricos: =((45-22.5)^2-(70000-32500)^2)^(1/2)=37500.00675 | Datos categóricos: (libros, comida)=1 + (nuevo, fiel)=1  Total=2 |
| Distancia total: 16667.6732 | | Distancia Total: 37502.00675 | |

1. Al objeto 4 se le asigna el valor de **K1**
2. Se recalcula la distancia entre los objetos y los nuevos prototipos, si el objeto presenta una menor distancia el objeto se mueve del prototipo asignado originalmente al prototipo que presente la menor distancia

Tabla 15 Iteración 2 distancia objeto 5

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 5** con el prototipo **K1** | | Distancia del **objeto 5** con el prototipo **K2** | |
| Datos numéricos: =((23-34.67)^2-(35000-32500)^2)^(1/2)= 18333.33371 | Datos categóricos: (electrónica, ropa)= 1+ (fiel, nuevo)= 1  Total=2 | Datos numéricos: =((23-22.5)^2-(35000-32500)^2)^(1/2)=2500.00005 | Datos categóricos: (electrónica, comida)=1 + (fiel, fiel)=0  Total=1 |
| Distancia total: 18335.33371 | | Distancia Total: 2501.00005 | |

1. Al objeto 5 se le asigna el valor de **K2**
2. Todos los objetos han sido asignados en un prototipo como se aprecia en la tabla 16

Tabla 16 Asignación de objetos segunda iteración

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cliente | Edad | Ingresos Anuales ($) | Categoría de Producto Favorito | Estado del cliente | Prototipo |
| 1 | 25 | 40,000 | Electrónica | nuevo | K2 |
| 2 | 34 | 50,000 | Ropa | regular | K1 |
| 3 | 22 | 30,000 | Comida | fiel | K2 |
| 4 | 45 | 70,000 | Libros | nuevo | K1 |
| 5 | 23 | 35,000 | Electrónica | fiel | K2 |

1. Se asignan los nuevos prototipos, para cada prototipo se calcula el promedio de los atributos numéricos y a través de la medida de similaridad y la moda para los atributos categóricos como se muestra en las tablas 9 y 10

Tabla 17 Reasignación de objetos, grupo 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cliente | Edad | Ingresos Anuales ($) | Categoría de Producto Favorito | Estado del cliente | Prototipo |
| 1 | 25 | 40,000 | Electrónica | nuevo | K2 |
| 3 | 22 | 30,000 | Comida | fiel | K2 |
| 5 | 23 | 35,000 | Electrónica | fiel | K2 |
| Promedio | 23.33 | 35000 | Electrónica | Fiel | K2 |

Tabla 18 Reasignación de objetos, grupo 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cliente | Edad | Ingresos Anuales ($) | Categoría de Producto Favorito | Estado del cliente | Prototipo |
| 2 | 34 | 50,000 | Ropa | regular | K1 |
| 4 | 45 | 70,000 | Libros | nuevo | K1 |
| Promedio | 39.5 | 60000 | Ropa | REGULAR | K1 |

1. Se recalcula la distancia entre los objetos y los nuevos prototipos, si el objeto presenta una menor distancia el objeto se mueve del prototipo asignado originalmente al prototipo que presente la menor distancia

Tabla 19 Iteración 3 distancia objeto 1

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 1** con el prototipo **K1** | | Distancia del **objeto 1** con el prototipo **K2** | |
| Datos numéricos: =((25-39.5)^2-(40000-60000)^2)^(1/2)= 20000.00526 | Datos categóricos: (electrónica, ropa)= 1+ (nuevo, regular)= 1  Total=2 | Datos numéricos: =((25-23.33)^2-(40000-35000)^2)^(1/2)=5000.000279 | Datos categóricos: (electrónica, electrónica)=0+ (nuevo, fiel)=1  Total=1 |
| Distancia total: 20002.00526 | | Distancia Total: 5001.000279 | |

1. Al objeto 1 se le asigna el valor de **K2**
2. Se recalcula la distancia entre los objetos y los nuevos prototipos, si el objeto presenta una menor distancia el objeto se mueve del prototipo asignado originalmente al prototipo que presente la menor distancia

Tabla 20 Iteración 3 distancia objeto 2

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 2** con el prototipo **K1** | | Distancia del **objeto 2** con el prototipo **K2** | |
| Datos numéricos: =((34-39.5)^2-(50000-60000)^2)^(1/2)= 10000.00151 | Datos categóricos: (ropa, ropa)= 0+ (regular, regular)= 0  Total=0 | Datos numéricos: =((34-23.33)^2-(50000-32500)^2)^(1/2)= 15000.00379 | Datos categóricos: (ropa, electrónica)=1 + (regular, fiel)=1  Total=2 |
| Distancia total: 10000.00151 | | Distancia Total: 15002.00379 | |

1. Al objeto 2 se le asigna el valor de **K1**
2. Se recalcula la distancia entre los objetos y los nuevos prototipos, si el objeto presenta una menor distancia el objeto se mueve del prototipo asignado originalmente al prototipo que presente la menor distancia

Tabla 21 Iteración 3 distancia objeto 3

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 3** con el prototipo **K1** | | Distancia del **objeto 3** con el prototipo **K2** | |
| Datos numéricos: =((22-39.5)^2-(30000-60000)^2)^(1/2)= 30000.0051 | Datos categóricos: (comidas, ropa)= 1+ (fiel, regular)= 1  Total=2 | Datos numéricos: =((22-23.33)^2-(30000-32500)^2)^(1/2)= 5000.000177 | Datos categóricos: (comida, electrónica)=1 + (fiel, fiel)=0  Total=1 |
| Distancia total: 30002.0051 | | Distancia Total: 5001.000177 | |

1. Al objeto 3 se le asigna el valor de **K2**
2. Se recalcula la distancia entre los objetos y los nuevos prototipos, si el objeto presenta una menor distancia el objeto se mueve del prototipo asignado originalmente al prototipo que presente la menor distancia

Tabla 22 Iteración 3 distancia objeto 4

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 4** con el prototipo **K1** | | Distancia del **objeto 4** con el prototipo **K2** | |
| Datos numéricos: =((45-39,5)^2-(70000-60000)^2)^(1/2)= 10000.00151 | Datos categóricos: (libros, ropa)= 1+ (nuevo, regular)= 1  Total=2 | Datos numéricos: =((45-23.3)^2-(70000-35000)^2)^(1/2)= 35000.00671 | Datos categóricos: (libros, electrónica)=1 + (nuevo, fiel)=1  Total=2 |
| Distancia total: 10002.00151 | | Distancia Total: 35002.00671 | |

1. Al objeto 4 se le asigna el valor de **K1**
2. Se recalcula la distancia entre los objetos y los nuevos prototipos, si el objeto presenta una menor distancia el objeto se mueve del prototipo asignado originalmente al prototipo que presente la menor distancia

Tabla 23 Iteración 3 distancia objeto 5

|  |  |  |  |
| --- | --- | --- | --- |
| Distancia del **objeto 5** con el prototipo **K1** | | Distancia del **objeto 5** con el prototipo **K2** | |
| Datos numéricos: =((23-39.5)^2-(35000-60000)^2)^(1/2)= 25000.00544 | Datos categóricos: (electrónica, ropa)= 1+ (fiel, regular)= 1  Total=2 | Datos numéricos: =((23-23.3)^2-(35000-32500)^2)^(1/2)= 0.33 | Datos categóricos: (electrónica, electrónica) =0 + (fiel, fiel)=0  Total=0 |
| Distancia total: 25002.00544 | | Distancia Total: 0.33 | |

1. Al objeto 5 se le asigna el valor de **K2**
2. Todos los objetos han sido asignados en un prototipo como se aprecia en la tabla 16

Tabla 24 Asignación de objetos tercera iteración

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cliente | Edad | Ingresos Anuales ($) | Categoría de Producto Favorito | Estado del cliente | Prototipo |
| 1 | 25 | 40,000 | Electrónica | nuevo | K2 |
| 2 | 34 | 50,000 | Ropa | regular | K1 |
| 3 | 22 | 30,000 | Comida | fiel | K2 |
| 4 | 45 | 70,000 | Libros | nuevo | K1 |
| 5 | 23 | 35,000 | Electrónica | fiel | K2 |

El algoritmo realiza el mismo procedimiento hasta que converja, en este caso converge debido a que ya no hubo movimientos, es decir, que movimiento=0.

## Investigaciones y proyectos usados con el algoritmo K-Prototypes

### Mejora de K-Prototypes mediante optimización evolutiva

Investigadores han propuesto mejoras al algoritmo K-Prototypes usando técnicas de optimización evolutiva como los algoritmos genéticos. Estos métodos buscan optimizar la selección de prototipos iniciales y el parámetro 𝛾 γ, mejorando la precisión y estabilidad del clustering.

### Aplicaciones en bioinformática

En el campo de la bioinformática, el K-Prototypes se ha utilizado para agrupar datos genéticos y clínicos, donde los atributos incluyen información tanto numérica (como la expresión génica) como categórica (como el tipo de enfermedad).

### Optimización del Transporte Público en Singapur

Contexto: En un estudio realizado por la Universidad Nacional de Singapur, K-Prototypes se utilizó para analizar datos de transporte público, incluyendo tiempos de viaje (numéricos) y tipos de usuarios (categóricos).

Resultados: La investigación ayudó a mejorar la planificación de rutas y la distribución de recursos en el transporte público.

### Segmentación de Clientes en Comercio Electrónico en China

Contexto: Una gran empresa de comercio electrónico en China utilizó K-Prototypes para segmentar a sus clientes basándose en el historial de compras y preferencias de productos.

Resultados: La segmentación permitió personalizar las ofertas y mejorar la experiencia del cliente, resultando en un aumento significativo en la satisfacción y las tasas de conversión de ventas.

### Análisis de Datos Clínicos en Estados Unidos

Contexto: Un hospital en Estados Unidos aplicó K-Prototypes para analizar datos clínicos de pacientes, incluyendo datos numéricos (edad, resultados de pruebas) y categóricos (diagnósticos, tratamientos).

Resultados: Los clusters formados ayudaron a identificar patrones en los datos clínicos, lo cual mejoró la personalización de tratamientos y estrategias de atención.

## Herramientas utilizadas

**Visual Studio Code (VS Code)**

Visual Studio Code es un editor de código desarrollado por Microsoft que, a pesar de su ligereza, ofrece una gran potencia. Incluye soporte para la depuración, control de versiones Git, resaltado de sintaxis, IntelliSense (autocompletado inteligente), snippets, y refactorización de código. Su diseño extensible permite añadir funcionalidades a través de extensiones, haciéndolo altamente adaptable a diferentes lenguajes y necesidades de desarrollo. Una de sus características más útiles es la terminal integrada, que facilita la ejecución de comandos durante el desarrollo de software. Visual Studio Code no está restringido en cuanto al tipo de aplicaciones que se pueden desarrollar con él, ya que se adapta de manera flexible a las necesidades de cada proyecto. Estas características lo convierten en una opción preferida para entornos que requieren la integración de lenguajes y herramientas para el análisis de datos multivariantes.

**Python**

Python es un lenguaje de programación interpretado, de alto nivel y con un propósito general, que destaca por su diseño enfocado en la claridad y legibilidad del código, utilizando una sintaxis sencilla y significativa. Su popularidad se debe en gran parte a su facilidad de aprendizaje y a la vasta comunidad de desarrolladores que ofrece numerosos recursos y bibliotecas para diversas áreas, como el análisis de datos, el desarrollo web, la inteligencia artificial, entre otros.

Python fue creado por Guido Van Rossum, un programador originario de los Países Bajos. El desarrollo de Python comenzó en 1989 en el Centrum Wiskunde & Informatica (CWI) como un proyecto personal que Van Rossum emprendió durante las vacaciones navideñas. El nombre "Python" proviene del programa de televisión "Monty Python's Flying Circus", un show de la BBC del que Van Rossum era un gran admirador.

**Evolución de Python**

La primera versión de Python, la 0.9.0, fue lanzada por Guido Van Rossum en 1991 e incluía funcionalidades importantes como varios tipos de datos y mecanismos para la gestión de errores. En 1994, Python 1.0 se presentó con mejoras significativas en la manipulación de listas, facilitando operaciones como la asignación, el filtrado y la reducción. El 16 de octubre de 2000, Python 2.0 salió al público, introduciendo características útiles como el soporte para Unicode y una forma más compacta de iterar sobre listas. Python 3.0, lanzado el 3 de diciembre de 2008, trajo consigo mejoras como una nueva función de impresión y un soporte mejorado para la división de números y la gestión de errores.

**Librerías de Python**

Para el sistema se necesitará varias bibliotecas de Python que ayudarán en diferentes aspectos del desarrollo.

**Math**

Es una biblioteca estándar de Python que proporciona funciones matemáticas básicas. Esta librería incluye una variedad de funciones para realizar cálculos matemáticos comunes, como operaciones trigonométricas, exponenciales, logarítmicas, y más. También contiene constantes matemáticas como π (math.pi) y e (math.e).

**Random**

Biblioteca estándar de Python. Se utiliza para generar números aleatorios y realizar operaciones relacionadas con la aleatoriedad, como seleccionar elementos al azar de una lista, barajar secuencias, y más.

# Procedimiento y descripción de las actividades realizadas.

## Requisitos funcionales

Los Requisitos Funcionales describen lo que el sistema debe hacer, es decir, las funciones y características específicas que debe tener para cumplir con las necesidades del usuario.

Tabla 25 Requisitos funcionales

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Requerimiento | Descripción | Prioridad |
| RF1 | Capacidad para cargar cualquier archivo con estructura csv | El sistema permitirá cargar archivos con extensión csv, txt, xml, etc siempre y cuando el archivo este delimitado por comas. | ALTA |
| RF2 | Importar y procesar datos mixtos (numéricos y categóricos) | El sistema nos permitirá la carga de datos que contenga atributos numéricos como categóricos. Además de que nos permitirá identificar y separar los atributos numéricos de los categóricos | ALTA |
| RF3 | Elegir cuántos k se pueden generar | El sistema debe permitir al usuario especificar el número de clusters (K) para la ejecución del algoritmo. | MEDIA |
| RF4 | Inicialización aleatoria de prototipos | El sistema seleccionará aleatoriamente los K prototipos iniciales para empezar el proceso. | MEDIA |
| RF5 | Cálculo de distancias mixtas y numéricas | El sistema debe implementar la fórmula de distancia mixta que combina la distancia euclidiana para los atributos numéricos y la distancia de coincidencia para los atributos categóricos. | ALTA |
| RF6 | Asignación de objetos a prototipos | El sistema debe calcular la distancia entre cada objeto y los prototipos y asignar el objeto al cluster del prototipo más cercano. | ALTA |
| RF7 | Actualizar prototipos | El sistema debe recalcular los prototipos de cada cluster en cada iteración, actualizando los valores numéricos con promedios y los valores categóricos con la moda. | MEDIA |
| RF8 | Iterar hasta que haya convergencia | El sistema debe repetir el proceso de asignación y actualización de prototipos hasta que las asignaciones de cluster no cambien o hasta que se alcance un número máximo de iteraciones. | ALTA |
| RF9 | Visualización de resultados | Se permitirá la visualización de los resultados proporcionando un informe con los clusters resultantes, los prototipos finales y la asignación de objetos a cada cluster, el despliegue de los resultados será en tablas. | BAJA |
| RF10 | Guardar los resultados en carpetas | Los clusters resultantes se guardarán en una carpeta, dentro de dicha se formarán carpetas que contendrán los prototipos finales. | MEDIA |

## Requisitos no funcionales

Los Requisitos No Funcionales describen cómo el sistema debe comportarse y sus cualidades, en lugar de describir funciones específicas. Estos requisitos se centran en la calidad del sistema.

Tabla 26 Requisitos no funcionales

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Requerimiento | Descripción | Prioridad |
| RNF1 | Generación automática de tablas en sqlite | Se genera una tabla en automático en una base de datos sqlite partiendo de un archivo con estructura csv | BAJA |
| RNF2 | Adición de campo id si el archivo no cuenta con el | Si el archivo no cuenta con ID el sistema en automático lo genera. | ALTA |
| RNF3 | Eliminar campo ID | Los datos del campo ID no serán tomados en cuenta para el cálculo de los prototipos. | ALTA |
| RNF4 | Despliegue de las colecciones k dependiendo del botón que se desee visualizar | Generación de front-end dinámico de acuerdo del numero de K centros elegidos por el usuario | MEDIA |
| RNF5 | Eliminación de los registros k iniciales | El algoritmo no se compara con el mismo. | ALTA |
| RNF6 | Portabilidad | Es fácilmente portable para que pueda ejecutarse en diferentes entornos | MEDIA |
| RNF7 | Interfaz intuitiva | Las interfaces de usuario deberán de ser  entendibles, amigables e intuitivas. | BAJA |
| RNF8 | Reconstrucción de nombres y headers | Refactorización de nombres y headers para la inserción en las tablas sql tomando en cuenta que todos los caracteres especiales se van a cambiar por guiones bajos. | ALTA |

## Análisis de la base de datos

## Diseño y prototipado

### Diagrama de casos de uso

Los diagramas de casos de uso son una herramienta utilizada para representar los

requisitos y funcionalidades de un sistema, centrándose en las interacciones entre

los usuarios y el sistema. En la Figura 5 se presentan los diagramas de casos de

uso desarrollado para el sistema

Diagrama

Descripción generada automáticamente

Figura 5 Diagrama de casos de uso

### Diagrama de secuencias

El diagrama de secuencias es una herramienta visual que permite entender la interacción secuencial entre los distintos componentes de un sistema. En la Figura 34 se presentan el diagrama de secuencias desarrollado.

Diagrama

Descripción generada automáticamente

Figura 6 Diagrama de secuencias

### Diagrama de clases

## Desarrollo del sistema

El proyecto está compuesto por las siguientes carpetas y scrips de Python

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

Figura 7 Estructura de archivos del proyecto

La carpeta DOCUMENTOS contiene todos los archivos que se quieran exportar a la base de datos SQLite y se quieran usar en el algoritmo K-Prototype (puede ser de cualquier extensión, pero debe tener estructura CSV), dentro de esta carpeta esta otra carpeta llamada COLLECTIONS, en la cual se van a guardar las carpetas y documentos de cada iteración del programa

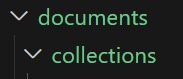


Figura 8 Carpeta documents y collections

Dentro de esta carpeta hay varias subcarpetas las cuales se generan automáticamente en cada ejecución del programa, estas carpetas están nombradas con una sintaxis especifica la cual es la siguiente:

**DATOS\_NO.BDD\_NO.PROTOTIPOS\_FECHA\_HORA**

De esta forma se garantiza que siempre habrá una carpeta distinta

Texto

Descripción generada automáticamente con confianza media

Figura 9 Visualización de carpetas con iteraciones finales

Dentro de estas carpetas se encuentran archivos tipo TXT, los cuales contiene los resultados finales de los números de prototipos respectivamente

Texto

Descripción generada automáticamente

Figura 10 Archivos txt con resultados finales

Cada archivo esta formado de la siguiente forma:

Texto

Descripción generada automáticamente

Figura 11 Estructura de carpetas

La nomenclatura del archivo se refiere al numero de la base de datos y el número del K dentro del archivo la primera sección están los prototipos finales de esa iteración y abajo están las colecciones de igual forma están seccionadas por K

El siguiente script es app.py el cual es el route de la aplicación, esto se refiere a que, por este script, levantamos el servidor y los end-points de la aplicación por medio de estos vamos a poder ingresar a la página.

* El init es la pagina inicial de desarrollo, el cual nos pide el numero de K y tiene un botón para llamar al método K-Prototype
* El k-prototypes es la ruta la cual tiene toda la lógica del desarrollo el cual detona el script controllerK\_prototipe.py

Texto

Descripción generada automáticamente

Figura 12 Levantar los datos

El script const.py tiene la constante full\_table\_name el cual es el nombre del archivo CSV que vamos a utilizar en esa ejecución, es importante tomar en cuenta que siempre se le debe poner la extension. Cabe recalcar que en este mismo script existe una función llamada limpiar\_nombre la cual reemplaza los caracteres especiales del nombre del archivo por guiones bajos para no tener problemas al momento de la ejecución.

Texto

Descripción generada automáticamente

Figura 13 Importar archivo

El script de controllerK\_Prototipe es el script maestro del algoritmo k-Prototype En este script se realiza lo siguiente:

* Clase inicial que llama a todo el sistema
* Se leen los archivos para posteriormente subirse a la bdd SQLite
* Se definen el número de K-Prototipos
* Se obtiene los números aleatorios delimitado por el minimo y máximo por los registros de base de datos, de esos números random se obtienen los datos
* Ejecución del algoritmo K-Prototype, mandando como atributos de las K’s
* Se guardan los datos de las iteraciones

Texto

Descripción generada automáticamente

Figura 14 Clase inicial maestra

Database.db es un archivo de tipo binario el cual no se puede abrir con un editor normal es necesario utilizar un visualizador para tipos de archivos .db, ejemplos: DB Browser, DBeaver.



Figura 15 Base de datos

DatabaseController.py este script administra toda la base de datos SQLite (database.db).

* Este método es de los mas importantes de este script, es el administrador de sesión de la bdd, el cual exportamos a otros métodos y scripts en todo el proyecto, en concreto la conexión y el cursor (hacer acciones en la BDD)

Texto

Descripción generada automáticamente

Figura 16 Administrador de sesión de la bdd

* En este método obtenemos el número máximo de registros de la tabla para posteriormente en este rango de números escoger k números aleatorios, entra a un ciclo para escoger los k prototipos aleatoriamente limitando que no sean iguales los k prototipos

Texto

Descripción generada automáticamente

Figura 17 Registros de la tabla

getDataPrototipes de los números aleatorios que se obtuvieron en el método anterior obtenemos los datos de estos

Texto

Descripción generada automáticamente

Figura 18 Obtención números aleatorios

getAllDatabase obtener toda la tabla para poder aplicar el algoritmo en estos datos, discriminando los K que se eligieron aleatoriamente, para evitar compararse consigo mismos

Texto

Descripción generada automáticamente

Figura 19 Obtención de la tabla

getHeaderTable se obtienen los headers de la tabla delimitados por el método getHeaderTableSelect

Texto

Descripción generada automáticamente

Figura 20 Obtención de cabezales de cada tabla

getHeaderTableSelect hace una proyección de los campos de la tabla discriminando los campos que nosotros queramos en este caso el ID

Texto

Descripción generada automáticamente

Figura 21 Proyección de los campos de la tabla

Se usa en conjunto con el método getAllDataBase el cual nos ayuda a poner el los id que se van a discrimar

Interfaz de usuario gráfica, Texto, Sitio web

Descripción generada automáticamente

Figura 22 Discriminación de prototipos

GetIfAlreadyExists este método se usa para identificar si el archivo que se esta leyendo ya se ha leído anteriormente esto para controlar la sintaxis de los archivos que se van a guardar

Texto

Descripción generada automáticamente

Figura 23 Lectura de archivos

Si el archivo que se quiere analizar con k-prototype no se ha leído anteriormente se agrega a la tabla de la bdd tableCount, de esta forma podemos controlar la sintaxis de los archivos que se van a generar

Texto

Descripción generada automáticamente

Figura 24 Inserción de archivo

Nos regresa el numero o id de la tabla que se está consultando, esto sirve para controlar la sintaxis de los archivos que se van a generar.

Texto

Descripción generada automáticamente

Figura 25 Consulta de la tabla que se usa

**EL SIGUIENTE SCRIPT ES K\_PROTOTYPE.PY**

Este script es el contiene toda la lógica del k-prototype, tiene varias funciones, la primera es controller\_Kprototipe

Esta función es el orquestador de el k-prototype por lo tanto es el que decide si se hace una nueva iteración o no

Este script es el que controla todo el flujo del k prototype primero siempre hay una iteración que se ejecuta siempre sin importar nada después la segunda iteración se compara con la primera iteración para ver si se tiene que seguir iterando o no sí las iteraciones son mayores a 5000 entonces separa el proceso. La variable Total\_result guarda el resultado final de todas las iteraciones en un diccionario de Python donde este diccionario tiene las características ‘prototypes’ y ‘collections’ y se genera un nuevo diccionario por cada iteración y se agrega al arreglo total\_result

Texto

Descripción generada automáticamente

Figura 26 Iteraciones del K-Prototype

El siguiente método es el algoritmo como tal, donde hacemos todas las operaciones referentes al algoritmo

Texto

Descripción generada automáticamente

Figura 27 Operaciones del algoritmo K-Prototype

Este método se usa en el método anterior controller\_kprototipes el cual es un método de tipo boleano que identifica si hay movimientos, sino hay movimientos regresa un false, de lo contrario true

Texto

Descripción generada automáticamente

Figura 28 Checar e iterar

Este método genera los nuevos prototipos de cada iteración

Texto

Descripción generada automáticamente

Figura 29 Nuevos prototipos de cada iteración

**SCRIPT MANAGEDOCUMENTS.PY**

Este script tiene varias funciones, la primera:

Este método se ocupa para eliminar los caracteres especiales de la cabecera de los documentos

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

Figura 30 Método para eliminar caracteres especiales

Dsv\_to\_sqlite esta función exporta las bases de datos de cualquier tipo de archivo (con estructura csv) a una tabla en SQLite

Texto

Descripción generada automáticamente

Figura 31 Exporta las bases de datos de cualquier tipo de archivo

Esta función se encarga de guardar cuantos tipos de documentos sean necesarios por iteración, tomando en cuenta el número de k dado desde el inicio el numero de base de datos y los datos finales de las iteraciones

Texto

Descripción generada automáticamente

Figura 32 Documentos necesarios por iteración

**Archivo view.py**

Esta función nos muestra los resultados de la ejecución

Texto

Descripción generada automáticamente

Figura 33 Interfaz gráfica 1

Texto

Descripción generada automáticamente

Figura 34 Interfaz gráfica 2

Texto

Descripción generada automáticamente

Figura 35 Interfaz gráfica 3

Texto

Descripción generada automáticamente

Figura 36 Interfaz gráfica 4

Y se mostrara la siguiente vista

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Figura 37 Vista total de iteraciones

Muestra la ventana de inicio para inicializar el valor de las k y enviar la ejecución de k-prototype

Texto

Descripción generada automáticamente

Figura 38 Vista principal

Se vera de la siguiente manera

Imagen que contiene Texto

Descripción generada automáticamente

Figura 39 Vista inicialización de ventana principal

Nos mostrara el aviso de máximas iteraciones

Texto

Descripción generada automáticamente

Figura 40 Error de la base de datos

Se mostrará de la siguiente manera

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Figura 41 Aviso repetir proceso

## Etapa de pruebas

# Resultados

## Vista de usuario

Iniciamos subiendo los archivos a esta parte del programa:

Texto

Descripción generada automáticamente

Figura 42 Carpetas de carga de archivos

Dentro de la carpeta ‘ documents ‘ tenemos otra carpeta en donde se sube el archivo con los datos que se usara para probar el algoritmo

Texto

Descripción generada automáticamente

Figura 43 Carga de archivos

Una vez teniendo el archivo arriba podemos utilizar un motor de base de datos, en este caso se esta utilizando DBeaver para poder visualizar los datos del archivo.

* Para crear la conexión damos clic en SQLite

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

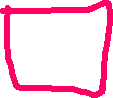


Figura 44 Conexión con la base de datos

* Buscamos la base de datos, le damos en siguiente y ya tenemos lista la conexión

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Figura 45 Path de la base de datos

* Tenemos lista nuestra conexión para poder visualizar los datos

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza media

Figura 46 Base de datos cargada a DBeaver

Una vez hecha la conexión podemos visualizar la tabla desde DBeaver, esta tabla se carga en automático una vez que agregamos el archivo desde el programa

Captura de pantalla de computadora

Descripción generada automáticamente con confianza media

Figura 47 Archivo cargado en DBeaver

Para correr el programa ejecutamos en consola la siguiente instrucción ‘ Python .app.py ’

Texto

Descripción generada automáticamente

Figura 48 Ejecución en consola del sistema

Posteriormente en el navegador entrar a la siguiente ruta



Figura 49 Ruta para entrar al proceso

Una vez ahí se nos mostrara la siguiente vista en donde se podrá elegir el numero de K-Centros que se deseen usar

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Figura 50 Interfaz inicial

Le damos clic en el botón enviar y podemos visualizar el número de iteraciones que se hicieron durante el proceso del algoritmo

Interfaz de usuario gráfica, Aplicación, Correo electrónico

Descripción generada automáticamente

Figura 51 Interfaz visualización número de iteraciones

Podemos visualizar los datos dando clic en cada uno de los botones

* Damos clic en Iteración #0

Escala de tiempo

Descripción generada automáticamente

Figura 52 Visualización datos de cada prototipo

* Para ver a que prototipo se fueron cada uno de los datos

Agrupamiento k0

Calendario

Descripción generada automáticamente con confianza media

Figura 53 Datos agrupamiento k0

Agrupamiento k1

Tabla

Descripción generada automáticamente con confianza media

Figura 54 Datos agrupamiento k1

Agrupamiento k2

Calendario

Descripción generada automáticamente

Figura 55 Datos agrupamiento k2

Esto se hace para visualizar las iteraciones obtenidas, posteriormente los datos serán almacenados en las carpetas que se encuentran dentro del sistema de forma automática

Para un mejor entendimiento los datos se guardan con la siguiente nomenclatura ‘ DATOS\_1\_NO.KPROTOTIPOS\_DIA\_FECHA ’ (se utilizó esa nomenclatura en caso de que se volviera a usar el archivo varias veces)

Texto, Calendario

Descripción generada automáticamente

Figura 56 Carpeta documents

Donde:

* DATOS\_1\_ : El “1” es el número de id que se le da al archivo una vez que se suba a la base de datos
* NO.KPROTOTIPOS: es el número de prototipos resultantes, cada carpeta contendrá archivos con extension txt que son los resultados finales del algoritmo al momento de converger

Pantalla de computadora con letras

Descripción generada automáticamente con confianza media

Figura 57 Carpeta collections

Dichos archivos los visualizamos al darle clic sobre ellos y se pueden visualizar de la siguiente manera

Texto

Descripción generada automáticamente

Figura 58 Archivos txt guardados en automático

# Conclusiones de Proyecto

Al inicio de este proyecto, el objetivo principal era implementar el algoritmo K-Prototypes para lograr una segmentación efectiva de datos mixtos, tanto numéricos como categóricos. A lo largo del desarrollo, se alcanzó este objetivo mediante la implementación de un código que permite agrupar datos de naturaleza diversa, cumpliendo con los criterios planteados al inicio. El algoritmo demostró ser capaz de gestionar volúmenes considerables de datos, proporcionando resultados que permiten identificar patrones y estructuras dentro del conjunto de datos.

Los resultados obtenidos evidencian que el K-Prototypes es una herramienta eficaz para la clasificación de datos mixtos, ya que permite generar clusters consistentes y bien definidos. Sin embargo, el desarrollo del proyecto también reveló algunas limitaciones. Una de las principales fue la complejidad en la selección de prototipos iniciales y el manejo de datos categóricos con múltiples categorías, lo que en algunos casos afectó la precisión de la segmentación. Otro desafío fue optimizar el tiempo de ejecución cuando se trataba de grandes volúmenes de datos, aunque se lograron implementar mejoras significativas para hacer el proceso más eficiente.

A pesar de estas limitaciones, el proyecto cumplió con sus objetivos y logró los resultados esperados. Además, el conocimiento adquirido durante la implementación del algoritmo, así como las habilidades técnicas desarrolladas, brindan una base sólida para futuras investigaciones y aplicaciones en áreas que involucren análisis de datos mixtos.

En conclusión, este proyecto no solo ha permitido implementar con éxito el algoritmo K-Prototypes, sino que también ha proporcionado valiosas lecciones sobre la manipulación de datos mixtos y la optimización de algoritmos, sentando las bases para una posible evolución en la forma en que se analizan datos heterogéneos en futuros trabajos.

# Recomendaciones

A partir de los resultados obtenidos y las limitaciones identificadas durante el desarrollo del proyecto, se proponen varias acciones para mejorar el algoritmo K-Prototypes y expandir su aplicación en futuros trabajos. Es fundamental optimizar la selección de prototipos iniciales mediante el uso de métodos más avanzados, como K-Means++ o algoritmos genéticos, lo que podría reducir el sesgo en la elección aleatoria y mejorar la calidad de los clusters desde el principio.

Asimismo, es importante profundizar en el manejo de datos categóricos que poseen múltiples categorías. Para ello, se sugiere explorar técnicas de codificación más sofisticadas, como la codificación de impacto, que podrían mejorar la precisión en el cálculo de distancias categóricas. Además, se recomienda continuar la investigación en la escalabilidad del algoritmo para garantizar su eficiencia al manejar grandes volúmenes de datos, considerando la posibilidad de aplicar técnicas de paralelización o recursos en la nube.

La evaluación de la calidad de los clusters generados es otro aspecto a tener en cuenta; incorporar métricas avanzadas, como la validación cruzada y el uso de índices específicos para datos mixtos, facilitará una mejor interpretación y validación de los resultados. También sería beneficioso aplicar el algoritmo K-Prototypes en diferentes contextos, como el análisis de comportamiento de clientes, datos médicos o redes sociales, para evaluar su adaptabilidad y utilidad en diversas áreas.

Por último, se sugiere trabajar en la automatización del proceso y en la creación de una versión del algoritmo que permita ajustes automáticos de parámetros, facilitando así su integración en entornos de análisis de datos en tiempo real. Estas recomendaciones tienen como objetivo impulsar la evolución del algoritmo K-Prototypes, asegurando su eficacia en el manejo de datos mixtos y fomentando su uso en diversas aplicaciones en el análisis de datos.

# Competencias desarrolladas y/o aplicadas

## Competencias Específicas

Durante el desarrollo del proyecto de implementación del algoritmo K-Prototypes, adquirí experiencia en diversas herramientas y tecnologías clave que fueron fundamentales para fortalecer mis conocimientos en análisis de datos y machine learning. Entre ellas, amplié mis habilidades en programación con Python, utilizando bibliotecas como pandas, numpy para manipular y analizar datos mixtos (numéricos y categóricos).

Asimismo, este proyecto me permitió consolidar mis conocimientos en técnicas de agrupamiento y segmentación de datos, específicamente en la aplicación del algoritmo K-Prototypes, lo cual implicó una comprensión profunda de las métricas de distancia y la forma en que se calculan las similitudes en conjuntos de datos heterogéneos. A lo largo de este proceso, también gané experiencia en la optimización de algoritmos, asegurando que el código fuera eficiente para manejar grandes volúmenes de datos.

El trabajo en este proyecto me proporcionó una visión más amplia sobre cómo implementar algoritmos de clustering desde cero, desde la inicialización de los prototipos hasta la asignación final de los objetos a los clusters, lo que me ha dado una sólida base para abordar problemas similares en futuros proyectos de análisis de datos.

## Competencias Genéricas

A lo largo del proyecto, mi capacidad de análisis fue fundamental, ya que tuve que estudiar y comprender cómo los diferentes tipos de datos interactúan dentro del algoritmo K-Prototypes. Esto me permitió desarrollar un enfoque más meticuloso en la preparación y limpieza de datos, asegurando que el proceso de agrupamiento fuera lo más preciso posible.

La gestión del tiempo fue otra competencia crucial que desarrollé. Establecer metas realistas y dividir el proyecto en etapas manejables me ayudó a cumplir con los plazos establecidos y a mantener un flujo de trabajo eficiente. Durante el proceso, también enfrenté desafíos técnicos que me obligaron a ser adaptable y a tomar decisiones rápidas para asegurar el progreso del proyecto, mejorando mi capacidad para enfrentar imprevistos y contratiempos.

Finalmente, este proyecto me brindó una mayor capacidad para trabajar de forma colaborativa, comunicando los avances y resultados de manera clara y efectiva, lo cual será valioso en futuros proyectos, donde la comunicación y la adaptación son esenciales para el éxito.

# Fuentes de información

López, E. S. (2006). Algoritmos de Agrupamiento en la Minería de Datos. Instituto Nacional de Astrofísica, Óptica y Electrónica. Recuperado de <https://inaoe.repositorioinstitucional.mx/jspui/bitstream/1009/628/1/LopezES.pdf>

Hernández, E. (2006). Tesis de Algoritmos de Clustering. Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional. Recuperado de <https://www.cs.cinvestav.mx/TesisGraduados/2006/tesisEdnaHernandez.pdf>

Amazon Web Services. (n.d.). What is Python? Recuperado de <https://aws.amazon.com/es/what-is/python/>

Jain, A. K. (2010). Data clustering: 50 years beyond K-means. Pattern Recognition Letters, 31(8), 651-666.

Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. IEEE Transactions on Neural Networks, 16(3), 645-678.

Everitt, B. S., Landau, S., Leese, M., & Stahl, D. (2011). Cluster Analysis. John Wiley & Sons.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... & Oliphant, T. E. (2020). Array programming with NumPy. Nature, 585(7825), 357-362. Recuperado de <https://numpy.org/>

McKinney, W. (2010). Data Structures for Statistical Computing in Python. Proceedings of the 9th Python in Science Conference, 51-56. Recuperado de <https://pandas.pydata.org/>

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830. Recuperado de <https://scikit-learn.org/>

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90-95. Recuperado de <https://matplotlib.org/>

Waskom, M. (2021). seaborn: statistical data visualization. Journal of Open Source Software, 6(60), 3021. Recuperado de <https://seaborn.pydata.org/>

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., ... & van der Walt, S. J. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. Nature Methods, 17(3), 261-272. Recuperado de <https://scipy.org/>

Ali, M., Jankowski, A., & Roy, A. (2020). PyCaret: An open source, low-code machine learning library in Python. arXiv preprint arXiv:2005.13547. Recuperado de <https://www.pycaret.org/>

Huang, Z. (1998). Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values. Data Mining and Knowledge Discovery, 2(3), 283-304.

Huang, Z. (1997). Clustering Large Data Sets with Mixed Numeric and Categorical Values. Proceedings of the First Pacific-Asia Conference on Knowledge Discovery and Data Mining, 21-34.

Siers, M. J., & Soria, E. M. (2017). Optimizing the K-Prototypes Clustering Algorithm Using Genetic Algorithms. Applied Soft Computing, 55, 196-207.

Wang, H., & Luan, Y. (2015). Clustering Mixed-Type Data Using K-Prototypes Algorithm. Journal of Bioinformatics and Computational Biology, 13(1), 1550004.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction. Springer Science & Business Media.

Zaki, M. J., & Meira Jr, W. (2014). Data mining and analysis: Fundamental concepts and algorithms. Cambridge University Press.

# Anexos

Carta de autorización por parte de la empresa u organización para la titulación y otros si son necesarios.

**FORMATO GENERAL**

1. Tipo de fuentes: Arial
2. Tamaño: 12 texto, 13 subtítulos y 14 títulos.
3. Alineación: Justificada (Referencias alineadas a la izquierda)
4. Márgenes: superior 2.5 cm. , inferior 2.5 cm., izq. 3.0 cm y der. 2.5 cm.
5. Paginación: abajo a la derecha a partir de la Introducción
6. Interlineado: 1.5
7. Las citas textuales si tienen cinco o más renglones, utilizar sangría
8. Títulos, subtítulos y encabezados no llevan puntuación
9. Utilizar esquema de numerado decimal, máximo tres niveles. Ejem.

**3 Marco Teórico**

**3.1 Bases de datos**

**3.1.1 Oracle**

1. Los Anexos se ordenan con letras mayúsculas de la A a la Z.
2. Encabezados: Los títulos y subtítulos en negritas.
3. Los títulos inician en una nueva hoja