eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea
BILBOKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE BILBAO

MÁSTER UNIVERSITARIO EN INGENIERÍA DE TELECOMUNICACIÓN

TRABAJO FIN DE MÁSTER

DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN WEB PARA LA VISUALIZACIÓN DE UNA PROPUESTA DE VALOR OBTENIDA MEDIANTE ANALÍTICA DE DATOS Y MACHINE LEARNING

Estudiante: Fernández León, Alicia
Directora: Huarte Arrayago, Maider
Codirectora: Astorga Burgo, Jasone
Curso académico: 2020-2021

Bilbao, 21 de Septiembre de 2021

Resumen

El crecimiento del volumen de datos ha provocado un cambio de estrategia a la hora de almacenar la información y hacer uso de ella. Mediante algoritmos de Machine Learning las empresas han migrado a un modelo de negocio basado en obtener propuestas de valor a partir de datos generados. Estas propuestas de valor se traducen en servicios para los clientes de las empresas.

En este trabajo se replica la estructura de una empresa orientada a datos. Haciendo uso de la información disponible en un lago de datos de usuarios y sus transacciones, respetando la privacidad de los mismos, se crean perfiles y se muestra una propuesta de valor a usuarios del sector de la banca. La propuesta de valor se obtiene haciendo uso de aprendizaje automático.

Palabras Clave: Aprendizaje automático, Compañía orientada a datos, Aplicación web, Lago de datos, Analítica de datos .

Datu bolumenaren hazkundeak estrategia aldaketa eragin du datuen laku batean eskuragarri dagoen informazioa gordetzeko eta hura erabiltzeko orduan. Makina ikasteko algoritmoen bidez, enpresek negozio eredua batera migratu dute sortutako datuetatik balio proposamenak lortzean oinarrituta. Balio proposamen hauek enpresen bezeroentzako zerbitzu bihurtzen dira.

Lan honetan datuetara bideratutako konpainiaren egitura errepikatzen da. Erabiltzaileen informazioa eta haien transakzioak erabiliz, haien pribatutasuna errespetatuz, profilak sortzen dira eta banku sektoreko erabiltzaileei balio proposamen bat erakusten zaie. Balio proposamena ikaskuntza automatikoa erabiliz lortzen da.

Gako-hitzak: Ikaskuntza automatikoa, Datuetara bideratutako enpresa, Web aplikazioa, Datuen lakua, Datuen analisia.

The growth in the volume of data has caused a change in strategy when it comes to storing information and making use of it. Through machine learning algorithms, companies have migrated to a business model based on obtaining value propositions from generated data. These value propositions are translated into services for the companies'clients.

In this project the structure of a data-oriented company is replicated. Making use of user information and their transactions available in a data lake, respecting their privacy, profiles are created and a value proposition is shown to users of the banking sector. The value proposition is obtained by making use of machine learning.

Keywords: Machine Learning, Data Driven Company, Web App, Data Lake, Data Analytics.

Índice

Lista de figuras

Lista de tablas

Lista de acrónimos

IDC International Data Corporation

IBM International Business Machines

RGPD Reglamento General de Protección de Datos

API Application Programming Interfaces

ADLS Azure Data Lake Store

AWS Amazon Web Services

JSON JavaScript Object Notation

BSON Binary JSON

CSV Comma Separated Values

NMI Normalized Mutual Information

WSGI Web Server Gateway Interface

WCSS Within-Cluster Sums of Squares

1. Introducción

Hoy en día estamos rodeados de datos. La llegada de nuevas tecnologías ha provocado que el volumen de información digital se haya disparado con un crecimiento exponencial diario. De acuerdo con el informe de IDC "The Digitization of the World From Edge to Core" [1], se prevee que para el año 2025 sea 175 veces mayor que en el año 2011.

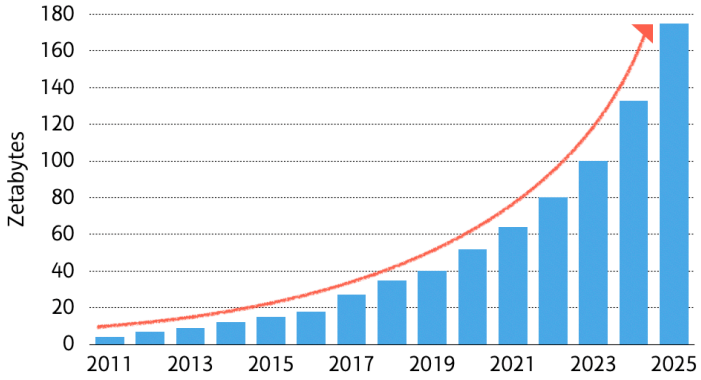


Figura 1: Crecimiento del volumen de datos por año.

En este contexto surge el término **Big Data**, el cual cobra impulso con la definición de las tres V del analista Doug Laney [2]: Volumen, Velocidad y Variedad. El término define esa gran cantidad de datos de diferentes fuentes que, debido a su magnitud y a la velocidad sin precedentes de generación, es difícil o imposible de procesar. El Volumen y la Variedad se plantean como un problema a la hora de realizar el almacenamiento de los datos, pero la llegada de herramientas compatibles con múltiples inquilinos como los **Data Lake** suponen un alivio en la carga. Por otro lado, otra poblemática que surge es el análisis inteligente de los mismos. De este punto se encarga el **Machine Learning**. Éste se aprovecha de la experiencia con el objetivo de proveer soluciones a ciertos problemas. Cualquier usuario que haya utilizado el cálculo de rutas de Google Maps o el sistema de recomendación de Netflix se ha beneficiado de éste area de la Inteligencia Artificial.

A pesar de la complicación del procesamiento de estos datos, se ha hecho un hueco en diferentes sectores profesionales hasta volverse una herramienta fundamental. Esto se debe a que el uso correcto de estos datos puede aportar un gran valor a usuarios y empresas. En este contexto surge el concepto de **Data Driven Companies**.

Este proyecto replica la arquitectura de una empresa orientada a datos del sector de la banca. Desde la construcción de un Data Lake hasta la analítica de los datos y segmentación de clientes para posterior visualización de esa propuesta de valor en una aplicación. Todas las partes esenciales para ser considerada una Data Driven Company. Es necesario avanzar hacia el cambio y adaptar las empresas al uso correcto de los datos para ofrecer nuevos servicios adaptados al usuario.

2. Contexto

En esta sección es necesario contextualizar dos términos mencionados anteriormente para poder desarrollar de forma correcta la solución. Por un lado, qué es una Data Driven Company y por otro lado, como se categorizan los algoritmos de Machine Learning y sus propiedades principales.

2.1. Data Driven Company

Para poder comprender la naturaleza del proyecto es necesario conocer que es una Data Driven Company y cuales son algunos de sus elementos esenciales, además de como se estructuran dentro de la entidad.

Una **Data Driven Company** es una entidad que dispone de información y que, de forma confiable, selecciona la parte que necesita de esta y mediante un equipo capacitado la transforma en una propuesta de valor. Hay empresas que nacen con esta filosofía, pero muchas otras pasan por diferentes etapas antes de convertirse en una.

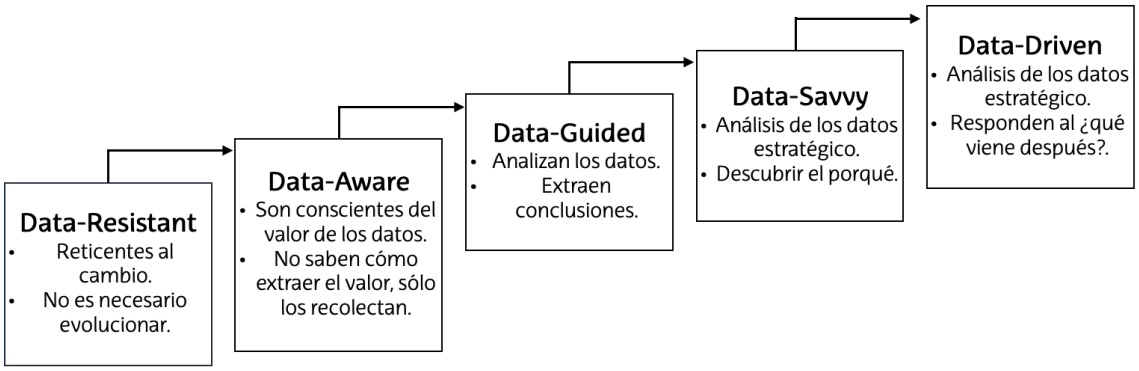


Figura 2: Etapas hasta convertirse en Data Driven Company.

En la Figura 2 se muestran las diferentes etapas y categorías existentes en el proceso de convertirse en una compañía orientada a los datos, siendo Data-Resistent aquellas que son reticentes al cambio. A partir de ella, según se aceptan diferentes puntos de esta nueva filosofía, las empresas se van acercando a convertirse en una Data Driven Company.

Y, ¿cómo se logra ser Data Driven Company? En la Figura 3 se muestra un ejemplo estructural de una, la cual se define de forma más concreta continuación.

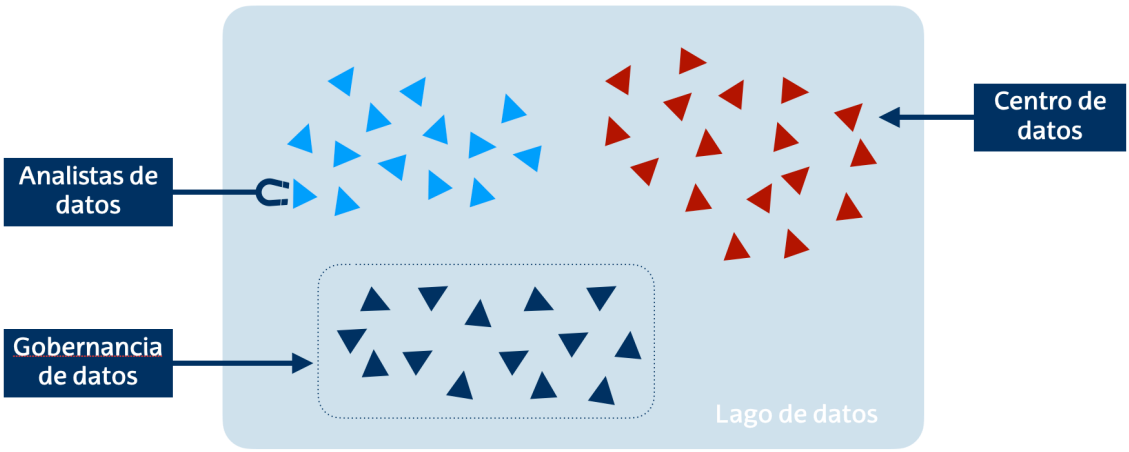


Figura 3: Estructura de Data Driven Company.

El primer paso es la creación de un **Data Lake** o lago de datos en el que se encuentra la información definida gracias a los namings, los triángulos de la Figura 3. Un Data Lake es un motor de procesamiento de consultas bajo demanda de múltiples inquilinos que permite utilizar un único lenguaje de consulta en los datos. Éste se abastece mediante el equipo de **Centro de datos**, el cual es el encargado de insertar los datos que le llegan desde diferentes fuentes origen garantizando un modelo de calidad sin redundancia.

Para que este lago funcione correctamente y debido a que la ingesta de los datos se puede realizar desde diferentes geografías, tiene que haber un encargado de gobernarlo, ahí es donde entran el grupo de **Gobernancia de datos**. Este equipo es el responsable de velar por:

- **Disponibilidad y trazabilidad**: el resto de las áreas de la entidad han de poder acceder a los datos democratizados y ha de ser posible el seguimiento de los mismos a través de los diferentes sistemas.
- **Contenido de calidad**: los datos han de estar correctamente definidos, sin redundancias y gestionados aplicando ciertas reglas y verificaciones.

Una de las principales funciones del gobierno de datos es la definición de los namings o nomenclaturas. De cara a la correcta extracción y análisis de los datos, es necesario que desde diferentes geografías realicen la ingesta de una forma común. Esto se ve mejor con un par de ejemplos.

El primer ejemplo está basado en los namings de la Figura 4 y la Tabla 1.

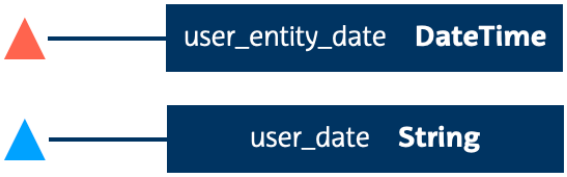


Figura 4: Namings de ejemplo.

Naming	Descripción	Formato
user_entity_date	Fecha de inscripción del usuario	**DateTime**
user_date	Fecha de inscripción del usuario	**String**

Tabla 1: Características de namings de la Figura 4

En la geofrafía A se inserta el dato de fecha de inscripción de usuario con formato lógico DateTime y en la geografía B con formato lógico String. Esto sucede porque se ha definido el mismo concepto mediante dos namings diferentes, lo cual provoca que a la hora de extraer los datos es necesario adaptarlos a un formato común para poder trabajarlos. Es una buena práctica disponer de un diccionario de namings sin redundacia, de forma que se compruebe la existencia de un naming que encaje con el dato a insertar y se sigan las reglas establecidas para el mismo. Este concepto se muestra en la Figura 5.

Figura 5: Solución a la primera problemática.

El segundo caso es aquel en el cual se define el mismo naming para diferentes conceptos en distintas geografías. Esto genera un error en el análisis de la información, ya que se puede estar tratando como si de lo mismo se tratara. Este ejemplo se ilustra en Tabla 2.

Naming	Descripción	Formato
user_entity_date	Fecha de inscripción del usuario	**DateTime**
user_entity_date	Fecha de baja del usuario	**DateTime**

Tabla 2: Características de namings de la Figura 5

La solución para este caso es, de nuevo, disponer de un diccionario de namings común para todas las geografías en el que se recojan los conceptos ya existentes. La solución se muestra en la Tabla 3 y en la Figura 6.

Naming	Descripción	Formato
user_inscription_date	Fecha de inscripción del usuario	**DateTime**
user_deregistration_date	Fecha de baja del usuario	**DateTime**

Tabla 3: Características de namings de la Figura 5

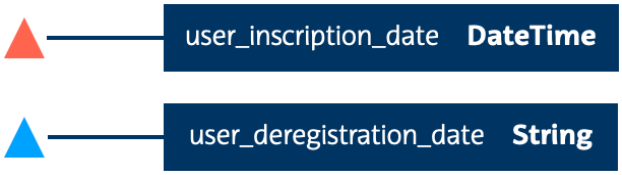


Figura 6: Solución a la segunda problemática.

Una vez el Data Lake se encuentra correctamente definido y funcionando, es momento de darle uso. De este punto se encargan los **Analistas de datos**. Este equipo se encarga de tomar los datos en bruto del mismo y generan propuestas de valor extraídas a partir de ellos haciendo uso de Machine Learning.

2.2. Machine Learning

Además de comprender que es una Data Driven Company, se considera necesario explicar y contextualizar el estado del arte del Machine Learning y más concretamente, de los algoritmos de clustering que se utilizan en el proyecto.

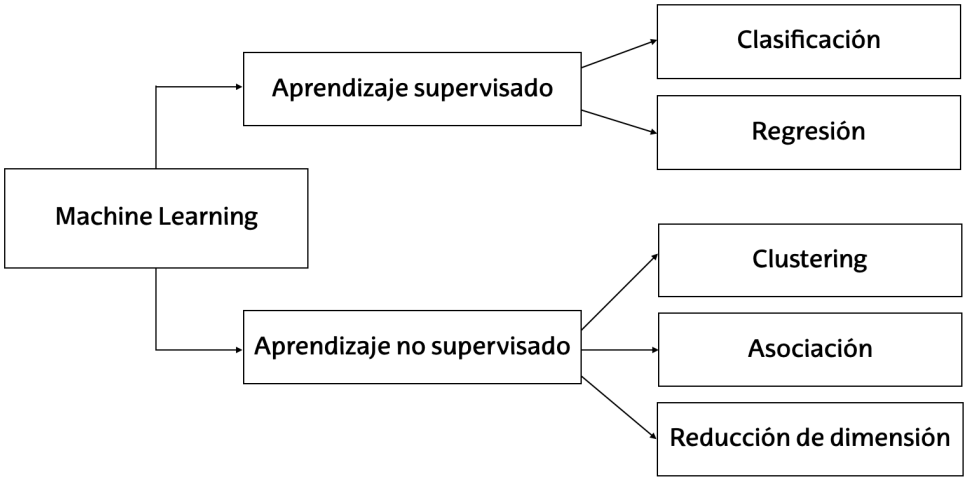


Figura 7: Clasificacion de algoritmos de Machine Learning.

De acuerdo con IBM [5] y la Figura 7, existen dos aproximaciones para dividir este concepto:

- **Aprendizaje Supervisado**: se define por el uso de conjuntos de datos etiquetados. Estos conjuntos, usados como entradas y salidas etiquetadas, se utilizan para entrenar algoritmos para clasificar o predecir resultados con precisión. Este aprendizaje se divide en dos tipos:
    - **Clasificación**: utilizan un algoritmo para asignar con precisión datos de prueba en categorías específicas. Un ejemplo real son los algoritmos usados para clasificar el spam en una carpeta separada de su bandeja de entrada.
    - **Regresión**: utiliza un algoritmo para comprender la relación entre las variables dependientes e independientes. Son útiles para predecir valores numéricos basados en diferentes puntos de datos, como las proyecciones de ingresos para una empresa determinada.

- **Aprendizaje no Supervisado**: utiliza algoritmos de aprendizaje automático para analizar y agrupar conjuntos de datos sin etiquetar. Estos algoritmos descubren patrones ocultos sin la necesidad de intervención humana, esto es lo que explica el concepto de "no supervisados". Este aprendizaje se divide en tres tipos:
    - **Clustering**: es una técnica de minería de datos para agrupar datos sin etiquetar en función de sus similitudes o diferencias. Esta técnica es útil, por ejemplo, para la segmentación del mercado.
    - **Asociación**: utiliza diferentes reglas para encontrar relaciones entre variables en un conjunto de datos determinado. Estos métodos se utilizan en los motores de recomendación en los comercios online.
    - **Reducción de dimensión**: se utiliza cuando el número de características en un conjunto de datos determinado es demasiado alto. Reduce la cantidad de entradas de datos a un tamaño manejable preservando la integridad de los datos. A menudo, esta técnica se utiliza en la etapa de preprocesamiento.

Dentro de todos tipos explicados anteriormente los que encajan en la necesidad del proyecto son los de clustering o agrupamiento. En los siguientes apartados se va a desarrollar los tipos de algoritmos existentes en este grupo y los pasos a realizar.

2.2.1. Tipos de algoritmos de clustering

Existen muchas características para definir los algoritmos de clustering, lo cual hace difícil realizar una clasificación de los mismos. Esto da lugar a, según la documentación consultada, encontrar diferentes clasificaciones. La forma más explicativa para definir estos tipos, disponible en el documento [7], es escoger cada característica y hacer divisiones naturales en función de estas:

- **Según si se conoce el número de clusters:**
    - **Jerárquico**. Se generan una serie de particiones estructuradas en una jerarquía. Se desconoce el número de clusters final.
        - Aglomerativos. Parten de que cada patrón forma un cluster.
        - Divisorios. Al contrario que en el caso anterior, parten de que existe un único cluster del que forman parte todos los patrones.
    - **Particional**. Se conoce el número de clusters final y el problema consiste en distribuir los patrones en los diferentes clusters maximizando una medida de similitud entre los mismos.
        - Agrupación exclusiva. Un patrón sólo puede pertenecer a un cluster.
        - Agrupación solapada. Un patrón puede pertenecer a más de un cluster.
- **Según como utilicen la función de distancia para agrupar clusters:**
    - **Enlace simple**. Se calcula la distancia entre dos clusters como la distancia mínima entre los pares de patrones, cogiendo uno por cluster.
    - **Enlace Completo**. Se calcula la distancia entre dos clusters como la máxima distancia entre los pares de patrones, cogiendo un patrón de cada cluster.
- **...**

Los tipos no tienen que ser de una subcategoría en concreto, pueden ser combinación. Un ejemplo de esto son los algoritmos jerárquicos simples. En el apartado de Descripción de la solución de desarrollará una comparativa de algunos de estos algoritmos escogiendo una de las características comentadas. Para ello se hará uso de esta división natural de los algoritmos de clustering realizada en este apartado.

2.2.2. Pasos de desarrollo de algoritmos de clustering

En este punto se desarrolla el procedimiento típico de estos algoritmos. Este procedimiento se trata de una aproximación, por lo que pueden existir variaciones en la metodología para el desarrollo del mismo.

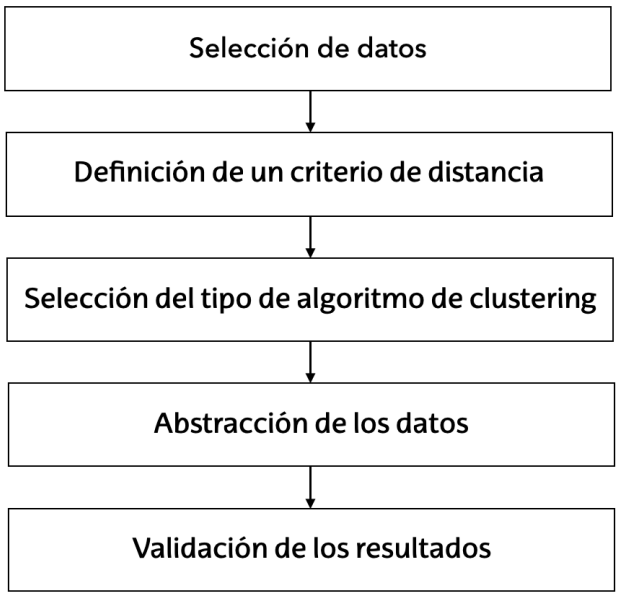


Figura 8: Pasos de algoritmo de clustering.

El procedimiento, disponible en la Figura 8, se divide en los siguientes pasos:

1. **Selección de datos**. En primer lugar hay que hacer una selección de las características escogiendo cual es el subconjunto más efectivo para poder realizar la agrupación de los mismos. En este punto también es necesario escalar estas variables para llevar los valores de diferente escala a una común. Esta normalización se puede realizar de diferentes maneras. Un par de ejemplos de esto son:
   - Normalización media: se obtiene restando la media y dividiendo por la desviación estándar para todos los elementos del conjunto de datos.
   - Normalización min-max: resta el valor mínimo del conjunto de datos y lo divide por la diferencia entre el valor más alto y el más bajo de la columna correspondiente.

2. **Definición de una medida de proximidad o criterio de distancia**. Esta selección se realiza en función de si los datos son cualitativos o cuantitativos:
    - Datos cualitativos: relativos a las cualidades. Este tipo de información está relacionada con los adjetivos.
    - Datos cuantitativos: relativos a los números. Todo aquello que se define como una cantidad.
3. **Selección del tipo de algoritmo de clustering**. Este punto consiste en escoger el algoritmo con el que se va a realizar el agrupamiento. Los tipos de algoritmos disponibles se explican en el punto anterior y la selección se realiza en función de la característica más relevante para el proyecto.
4. **Abstracción de los datos**. Extraer una muestra del conjunto de datos que defina una descripción compacta de diferentes clusters.
5. **Validación de los resultados**. Consiste en ver si los clusters obtenidos son correctos. Existen tres tipos de validación:
    - Externa: compara la estructura con una realizada a priori.
    - Interna: valida si la estructura obtenida es apropiada para los datos.
    - Compara dos estructuras y escoge cual es mejor.

3. Objetivos y alcance

3.1. Objetivo principal

En este proyecto el objetivo principal es obtener una propuesta de valor a partir del análisis de datos y Machine Learning, visualizando el resultado a través de una aplicación web. Concretamente, se ofrece al cliente de un banco recomendaciones de los gastos que se realizan en función de su perfil. Para ello, una vez organizados los datos y realizando una analítica de los mismos, mediante Machine Learning se obtiene esa propuesta de valor. Esta ha de estar disponible gracias a una herramienta de visualización.

3.2. Objetivos secundarios

En este apartado se definen los cuatros bloques principales de este proyecto. A partir de ahora, durante el resto del documento, el análisis se realizará en base a estos:

- Diseño de la solución. Diseño de los diferentes bloques funcionales que forman parte de la solución, desde la herramienta de almacenamiento al método de visualización de la propuesta de valor.
- Implementación de la solución. Implementación de los bloques diseñados y prueba del correcto funcionamiento:
    - Preparar un **repositorio de almacenamiento** que organice la información de forma óptima para el estudio de esta. Además de cumplir ciertas best practices y respetar el RGPD.
    - Desarrollo y comparativa de **algoritmos de clustering** que obtengan una propuesta de valor a partir de datos de clientes del banco. Una vez probados el que obtenga el mejor resultado basándonos en diferentes métricas será implementado en la solución final.
    - Desarrollo de una **API** a la que se conecta la herramienta de visualización para obtener la información relativa a los usuarios.
    - Desarrollar una **herramienta de visualización** para un cliente del sector de la banca. En la aplicación, por un lado se mostrará al usuario ciertos datos acerca del estado de su cuenta y de transacciones realizadas. Por otro lado, se mostrarán estadísticas obtenidas en base a su perfil.

Además de estos objetivos bloque, se preveen objetivos de cara al futuro como la actualización de la herramienta de visualización para ofrecer nuevas funcionalidades o la realización de modificaciones en el propio algoritmo con nuevos datos de análisis.

4. Beneficios

De cara a conocer el interés en el desarrollo del proyecto, se realiza un análisis de los beneficios que presenta. Este punto se centra en los beneficios técnicos, sociales y económicos. Todos ellos son considerados esenciales en una Data Driven Company.

4.1. Técnicos

Este punto resulta importante debido a la adaptación de la empresa al crecimiento del volumen de datos. Gracias al uso de un Data Lake se dispone de información de diferentes geografías cómodamente utilizando un lenguaje de base de datos común. Además de esto se realiza un cambio en la forma de estructurar la información, lo cual facilita el acceso a ella y el análisis de la misma. Esta forma de organizar los datos da lugar a propuestas de valor que se visualizan en una herramienta de visualización diseñada por la empresa.

4.2. Sociales

Este proyecto tiene como objetivo principal el análisis y agrupación de los usuarios según atributos en común. Esto supondrá una mejora, no sólo de cara a la propia empresa, tal y como se ha mencionado anteriormente, sino de cara a estos. Se trata de una propuesta de valor y una mejora en los servicios ofrecidos a cada uno de los usuarios. Gracias a la minería de los datos, se podrán extraer conclusiones que aporten información relevante para hacer propuestas de contrato o visualizar cuales son los gastos típicos según el perfil del usuario. En este punto resulta importante destacar también que la solución desarrollada se puede adaptar a otros contextos, no necesariamente al sector de la banca. Se ha escogido éste para disponer de un dataset y una base de datos a analizar.

4.3. Económicos

Este proyecto está orientado en gran parte al beneficio económico para las empresas que apliquen la solución. Aunque el objetivo principal sea la mejora de la estructura empresarial para poder adaptarse a la cantidad de datos generada en la actualidad, también ofrece un valor añadido a los usuarios. El cambio en el método de almacenamiento supone un ahorro económico y la mejora de los servicios al usuario supone adaptarse al mismo con el objetivo de ofrecer propuestas y aumentar así el número de clientes interesados en los servicios.

5. Análisis de alternativas

Para el análisis de alternativas se ha decidido que los criterios de selección y de ponderación sean los mismos para todas las comparativas. Esto se debe a que son adecuados en todos los casos. Los criterios y sus valores correspondientes son:

- **Sencillez (25 %)**. En este punto se define la facilidad en el uso de la alternativa, incluyendo el concepto de curva de aprendizaje.
- **Rendimiento (20 %)**. Capacidad ofrecida por la alternativa para obtener un resultado adecuado para el proyecto.
- **Soporte (20 %)**. Tiene en cuenta la documentación oficial disponible y el apoyo de la comunidad a la alternativa.
- **Costo (35 %)**. Incluye tanto el coste de licencia de uso como el coste asociado a aprender sobre la alternativa.

Para tomar la decisión final de qué opción planteada escoger, todos los valores se puntuan sobre diez y son ponderados según el porcentaje mencionado. En este caso se ha dado mucha relevancia al coste, es decir, a las opciones de desarrollo gratuitas debido a la naturaleza del proyecto. Esto puede variar considerablemente en situaciones reales.

5.1. Almacenamiento

Este es uno de los puntos principales del proyecto, ya que toda la información de la empresa de la cual se va a hacer uso se va a encontrar disponible ahí.

5.1.1. Método de almacenamiento

En este punto se han tenido en cuenta los formas de almacenamiento:

- **Almacén de datos tradicional**. Un almacén de datos es una combinación de tecnologías y componentes que permite recopilar y gestionar datos procedentes de diferentes orígenes para su uso.
- **Data Lake**. Se trata de un motor de procesamiento de consultas bajo demanda de múltiples inquilinos que permite utilizar un único lenguaje de consulta en los datos del almacén de objetos en la nube en múltiples formatos. El Data Lake es bastante diferente respecto al almacén de datos, ya que almacena información que no está lista para el consumo, sino que se recoge en estado natural.

Los dos puntos que definen al lago de datos, múltiples inquilinos con un único lenguaje de consulta común, son muy relevantes a la hora de escoger el método de almacenamiento y son los que le convierten en la opción ganadora. Además, tal y como se muestra en la Tabla 4, la opción de disponer de esta alternativa de forma gratuita ha supuesto la elección del Data Lake como solución más adecuada de cara al proyecto.

	Data Lake	Almacén de datos tradicional
Sencillez (25 %)	10	6
Rendimiento (20 %)	9	9
Soporte (20 %)	9	7
Costo (35 %)	10	9
Total	9.6	7.9

Tabla 4: Comparativa alternativas de almacenamiento.

5.1.2. Arquitectura

Una vez escogido el método de almacenamiento, es el momento de escoger los servicios y la arquitectura a utilizar. Las alternativas son:

- **MongoDB**. Atlas Data Lake es un motor de procesamiento de consultas bajo demanda de múltiples inquilinos que le permite utilizar el lenguaje de consulta MongoDB (MQL) en los datos del almacén de objetos en la nube en múltiples formatos.
- **Azure**. Azure es el Data Lake de Microsoft. Tiene una capa de almacenamiento Azure Data Lake Store o ADSL y otra de analítica con Azure Data Lake Analytics y HDInsight. Utiliza un lenguaje llamado U-SQL, que es una combinación de SQL y C#.

Dentro de las opciones de Data Lake se ha escogido MongoDB por la disponibilidad de recursos compartidos gratuitos para el desarrollo de la solución. Además de con sus propias bases de datos, también permite conectar con Amazon AWS S3. Se ha realizado un análisis, disponible en la Tabla 5, siguiendo los criterios definidos. En ésta se ve como Azure, a pesar de no ser la opción escogida, también supone una buena alternativa para el desarrollo.

	MongoDB	Azure
Sencillez (25 %)	10	9
Rendimiento (20 %)	10	10
Soporte (20 %)	9	9
Costo (35 %)	10	8
Total	9.8	8.9

Tabla 5: Comparativa de proveedores de servicios de almacenamiento.

5.2. Clustering

En este punto resulta interesante realizar un análisis de tanto el lenguaje utilizado como las librerías disponibles para realizar el clustering. Para esta elección no es necesario conocer el tipo de algoritmo de clustering que se va a utilizar, el análisis es común.

5.2.1. Lenguaje

Las alternativas principales respecto al lenguaje son:

- **Python**. Se considera el lider en lenguajes de desarrollo de Machine Learning debido a su simplicidad y facilidad de aprendizaje. Es un éxito entre los principiantes y viene con librerías como NumPy y Pandas.
- **R**. Este lenguaje está diseñado para análisis estadísticos y visualizaciones, se usa con frecuencia para detectar patrones en grandes bloques de datos. Los desarrolladores disponen de RStudio, su entorno de desarrollo gratuito.
- **Matlab**. Se considera el lenguaje de núcleo duro para matemáticos y científicos que se ocupan de sistemas complejos. Es rápido y estable.
- **Julia**. Lenguaje dinámico de alto nivel diseñado para abordar las necesidades del análisis numérico de alto rendimiento. Se integró con las mejores bibliotecas open source C y Fortran.

Teniendo en cuenta la Tabla 6 Python se presenta como la mejor opción de lenguaje para el desarrollo del algoritmo de clustering. Esto se debe a que presenta una buena curva de aprendizaje, se trata de un lenguaje de código abierto y ampliamente apoyado por la comunidad. En cuanto al rendimiento la mejor opción es R, pero no supone una diferencia tan notable para que sea el lenguaje escogido.

	Python	**R**	**Matlab**	**Julia**
Sencillez (25 %)	10	7	8	8
Rendimiento (20 %)	8	9	8	8
Soporte (20 %)	10	9	9	9
Costo (35 %)	10	10	5	10
Total	9.6	8.9	7.15	8.9

Tabla 6: Comparativa de lenguajes para clustering.

5.2.2. Librería

Las principales librerías de Machine Learning a analizar son:

- **Scikit Learn**. Librería de licencia libre en Python con herramientas simples y eficientes para el análisis predictivo de datos.
- **Scipy**. Es un ecosistema de software de código abierto basado en Python para matemáticas, ciencias e ingeniería.

Cualquiera de las alternativas disponible es correcta, pero se ha escogido Scikit Learn por la calidad de la documentación y la cantidad de algoritmos de clustering disponibles.

	Scikit Learn	Scipy
Sencillez (25 %)	9	8
Rendimiento (20 %)	9	8
Soporte (20 %)	10	7
Costo (35 %)	10	10
Total	9.6	8.5

Tabla 7: Comparativa de librerías de clustering.

Además de las librerías propias de clustering se hará uso de otras esenciales como Pandas y Numpy para el procesamiento de datos o Matplotlib para la visualización de los mismos. En estos casos no ha sido necesario realizar una comparativa ya que son consideradas todas ellas las esenciales.

5.3. API

Aunque es posible realizar el desarrollo de APIs desde cero, sea cual sea el lenguaje elegido, es conveniente partir de frameworks o plantillas. Por lo que en este punto se realiza un análisis tanto de los lenguajes como de los frameworks disponibles.

5.3.1. Lenguaje

Los lenguajes escogidos para el análisis de alternativas son:

- **Python**. Lenguaje de programación abierto multiparadigma. Soporta parcialmente la orientación a objetos y, en menor medida, programación funcional.
- **Ruby**. Lenguaje de programación orientado a objetos de código abierto y sintaxis natural.

	Python	Ruby
Sencillez (25 %)	10	9
Rendimiento (20 %)	8	8
Soporte (20 %)	10	8
Costo (35 %)	10	10
Total	9.6	9.2

Tabla 8: Comparativa de lenguajes para la API.

Como se puede observar el la Tabla 8 se escoge Python ya que se trata de un lenguaje con una sencilla curva de aprendizaje, de código libre y con múltiples librerías que facilitan el desarrollo de la API.

5.3.2. Framework

Las alternativas de framework disponibles para Python son:

- **Django**. Se trata de un framework de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como modelo–vista–controlador.
- **Flask**. Es un framework minimalista escrito en Python que permite crear aplicaciones web con un mínimo número de líneas de código rápidamente.

Flask es un web framework en Python que simplifica la manera de publicar nuestra propia API, ideal para el prototipado rápido. Se trata de una buena alternativa para quien empieza desde cero. Además, presenta una velocidad mayor que Django.

	Django	Flask
Sencillez (25 %)	7	9
Rendimiento (20 %)	8	9
Soporte (20 %)	9	9
Costo (35 %)	9	9
Total	8.3	9

Tabla 9: Comparativa de frameworks para la API.

5.4. Aplicación

En este punto resulta interesante analizar que tipo de herramienta de visualización se ha de desarrollar. Para ello, al igual que con la API, es necesario analizar tanto el lenguaje a utilizar como los frameworks y librerías disponibles para los mismos.

5.4.1. Tipo de aplicación

Los tipos de aplicación, disponibles en la Figura 9, que encajan en el proyecto son:

- **Aplicación web**. Las aplicaciones web son programas informáticos que se ejecutan en un servidor web, al que los usuarios acceden utilizando un programa de navegación por Internet. No necesitan instalarse en el dispositivo, pero sí precisan de conexión a la red.
- **Aplicación nativa**. Las aplicaciones nativas son aquellas aplicaciones desarrolladas para un dispositivo determinado, móvil u ordenador. Funcionan sin necesidad de ningún programa externo.

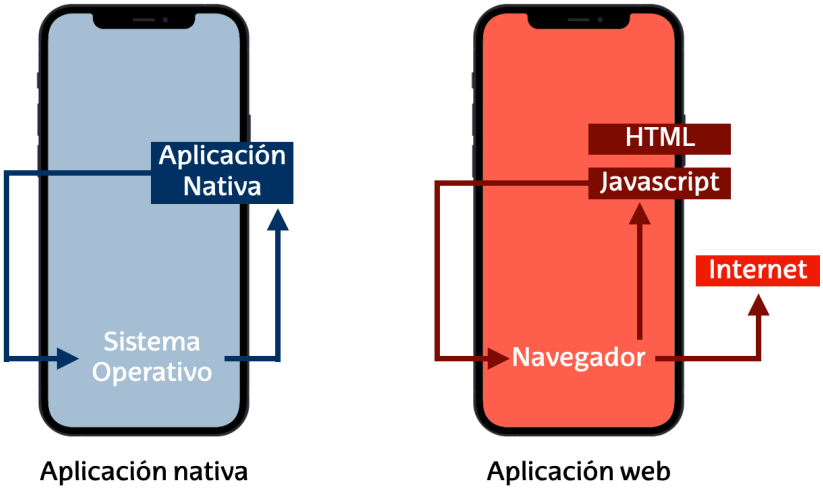


Figura 9: Alternativas para la aplicación.

En la Figura 9 se muestra la diferencia entre el funcionamiento de una aplicación nativa y una aplicación web. En el primer caso, se trata de una aplicación instalada en el dispositivo que aprovecha el 100 % del sistema operativo. En el segundo caso, se hace uso de la aplicación mediante un navegador instalado en el sistema operativo. Debido a que su uso es en navegador, los lenguajes detrás de su desarrollo son HTML, JavaScript y lenguajes de programación web.

Ambas opciones presentan una serie de ventajas y desventajas. En primer lugar, las aplicaciones web requieren de un solo desarrollo web mediante el cual se puede acceder a través de todos los tipos de dispositivos, independientemente del sistema operativo. Para el caso de nativas es necesario desarrollar para cada sistema operativo en concreto. Por otro lado, otro punto de interés relacionado con el caso anterior es que las aplicaciones nativas pueden aprovechar el 100 % de la funcionalidad ya que están programadas exclusivamente para dicho sistema operativo. A corto plazo y con el objetivo de tener una aplicación universal disponible para todos los dispositivos con un buen coste se ha decidido utilizar la primera opción. Esta decisión se ha tomado en base a los criterios estudiados en la Tabla 10.

	Aplicación web	Aplicación nativa
Sencillez (25 %)	9	7
Rendimiento (20 %)	7	9
Soporte (20 %)	10	7
Costo (35 %)	9	7
Total	8.8	7.4

Tabla 10: Comparativa de lenguaje de app web

5.4.2. Framework

Una vez escogido el tipo de aplicación y analizados los frameworks principales disponibles, se ha realizado un estudio disponible en la tabla 11. Como se puede observar cualquiera de las posibles soluciones encajan como opción para el desarrollo de la aplicación.

- **NextJS**. Framework construido sobre React.js que permite, instalando una sola dependencia, tener configurado todo lo que se necesita para crear una aplicación usando Babel, Webpack, server render, etc.
- **VueJS**. Framework basado en Javascript de código abierto para la construcción de interfaces de usuario y aplicaciones de una sola página.
- **Angular**. Framework de JavaScript de código abierto de Google que se utiliza para crear aplicaciones web de una sola página.

En la Tabla 11 se muestra como se escoge la opción de NextJs ya que el resultado es ligeramente mayor. La sencillez y el rendimiento se convierten en uno de los factores principales para esta elección. Por otro lado, permite trabajar con Tailwind. Tailwind es un CSS framework orientado al diseño de la aplicación web.

	NextJS	VueJS	Angular
Sencillez (25 %)	9	9	6
Rendimiento (20 %)	9	8	8
Soporte (20 %)	9	9	9
Costo (35 %)	10	10	10
Total	9.4	9.2	8.4

Tabla 11: Comparativa de lenguaje de app web

Las alternativas escogidas se encuentrandisponibles en la Figura 10. Estas se explicarán más en concreto en el apartado de Desarrollo de la solución.

Figura 10: Elección final de alternativas.

6. Análisis de riesgos

En este apartado se evalua los eventos que se pueden suceder impactando negativamente en el proyecto. Para la evaluación de estos riesgos se va a utilizar una metodología basada en tres pasos: identificación del riesgo, evaluación del mismo y elaboración de un plan de contingencia para reducir ese impacto en el caso de que suceda.

6.1. Identificación de los riesgos

Se ha realizado un estudio para identificar los posibles riesgos que pueden surgir en el proyecto:

- Error en el diseño
- Error en el presupuesto
- Superación de plazo planificado
- Mala gestión de los recursos
- Problemas técnicos

6.2. Evaluación de los riesgos

Para evaluar cada riesgo hay que tener en cuenta dos factores importantes, la probabilidad de que suceda y el impacto que tendría en nuestro proyecto. Esto se puede mostrar de forma visual con la ayuda de la matriz probabilidad-impacto de la Figura 11.

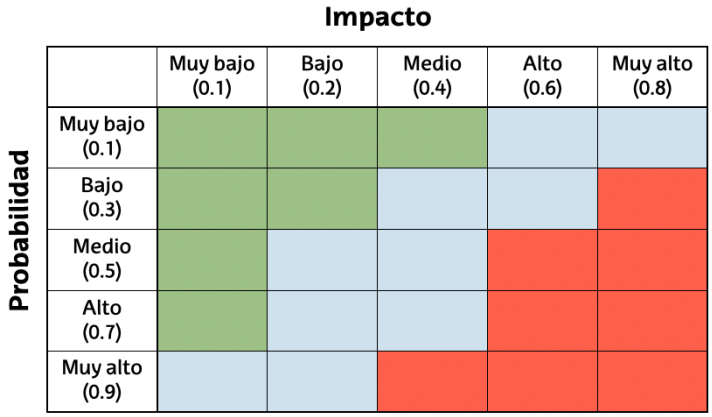


Figura 11: Matriz probabilidad impacto.

- **Error en el diseño (A)**:
  En este caso un mal diseño puede llevar a graves consecuencias en el caso de no ser detectado a tiempo. Un ejemplo de esto es un fallo en el diseño de la estructura de almacenamiento de datos puede dar lugar a la pérdida de los mismos. Esto resulta muy perjudicial y supone un gran impacto a nivel de entidad. En conjunto se considera que el impacto es alto. En cambio, la probabilidad de que ocurra es medio.

- **Error en el presupuesto (B)**:
  Se calcula el presupuesto de forma errónea porque no se tiene en cuenta algún factor o porque la previsión no es adecuada. En este caso, el impacto es medio, aunque dependerá de si el presupuesto se ha calculado por encima o por debajo del presupuesto real. Por otro lado, se ha considerado una probabilidad media de aparición.

- **Superación de plazo planificado (C)**:
  Esto se debe a que alguna de las tareas no se ha realizado en el tiempo asignado para ella. En este caso, el impacto será medio, ya que el proyecto puede disminuir su calidad. En cuanto a la probabilidad se considera baja, ya que se realiza un complejo estudio inicial para realizar la planificación.

- **Mala gestión de los recursos (D)**:
  La mala gestión de los recursos puede dar lugar a problemas en la planificación o en el presupuesto. El impacto es medio y la probabilidad de que ocurra es baja, ya que, al igual que en la planificación, se realiza un complejo estudio anterior.

- **Problemas técnicos (E)**;
  En este caso se plantean todo tipo de problemas técnicos que puedan surgir, tanto externos como internos. Un ejemplo de problema externo que puede surgir es que alguna herramienta de software libre deje de disponer de soporte. Dentro de las posibles problemáticas internas se encuentra la falta de integración entre las diferentes herramientas. Tanto el impacto como la probabilidad son altos.

En la Tabla 12 se encuentra disponible el valor numérico asignado a cada posible riesgo respecto a su probabilidad e impacto.

Riesgo	Probabilidad	Impacto	Resultado
A	0.5	0.6	0.3
B	0.5	0.4	0.2
C	0.3	0.4	0.12
D	0.1	0.4	0.04
E	0.7	0.8	0.56

Tabla 12: Tabla de riesgos.

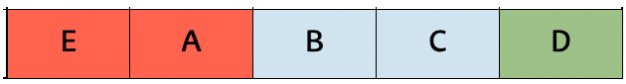


Figura 12: Orden de los riesgos según su probabilidad-impacto.

6.3. Plan de contingencia

Para realizar el plan de contingencia en primer lugar hay que ordenar los riesgos, figuras 12 y 13, en función del resultado de probabilidad-impacto obtenido en el paso anterior. De esta forma, es más sencillo evaluar el nivel de importancia que tienen y la necesidad de actuación sobre cada uno de ellos.

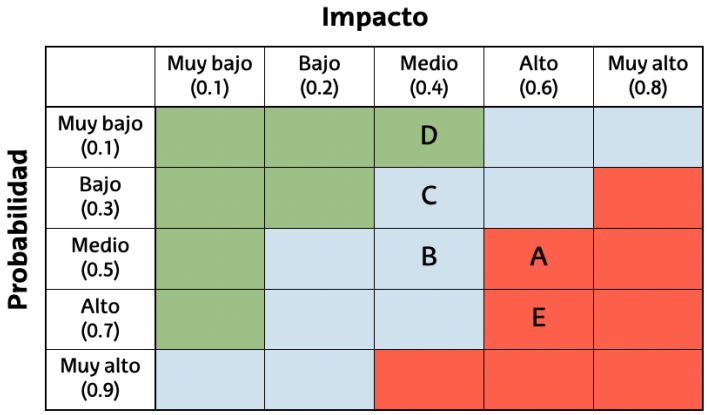


Figura 13: Matriz probabilidad impacto.

Los planes de contingencia se realizan para cada riesgo de forma individual y se centra en prevenirlo o en caso de que ocurra resolver la problemática lo antes posible disminuyendo el tiempo de indisponibilidad. Los planes de contingencia individuales en este proyecto son los siguientes.

- **Error en el diseño (A)**:
  Para evitar fallos, lo óptimo es la revisión ocasional del funcionamiento de los bloques de la entidad mediante fases de prueba de software para cada uno de ellos y pruebas globales una vez disponible el bloque funcional completo.
- **Error en el presupuesto (B)**:
  La forma óptima de prevenir errores en el presupuesto es tener en cuenta los posibles imprevistos añadiendo una partida en el presupuesto para ellos. Este punto es necesario debido a que es difícil predecir con exactitud las necesidades futuras que se plantean en el proyecto.
- **Superación de plazo planificado (C)**:
  La mejor forma de prevenir esto es realizar un estudio completo del proyecto de cara a la planificación y hacer revisiones semanales del estado de la fase en la que se encuentra el proyecto en ese momento mediante entregables puntuales.
- **Mala gestión de los recursos (D)**:
  Para evitar la mala gestión de los recursos y gastos injustificados se realiza la revisión mediante un libro de cuentas que permite llevar un control de gastos.
- **Problemas técnicos (E)**:
  Para evitar esta problemática se dispone de alternativas de software que encajen en el proyecto y se realiza seguimiento del estado de la herramienta y del funcionamiento de la misma.

7. Descripción de la solución

Para describir la solución se va a utilizar el esquema de la Figura 14. Este sigue el orden definido en los objetivos secundarios del proyecto.

1. **Diseño e implementación del Data Lake**. Almacen de datos de la empresa. En él se encuentran disponibles los datos de los usuarios que se utilizan para realizar el análisis y obtener valor del mismo.
2. **Desarrollo y elección del algoritmo de clustering**. Una vez los analistas de datos recogen la información del Data Lake y realizan el análisis de la misma, guardan la información de forma que queda accesible para la API. En el caso de este proyecto ese análisis consiste en la realización de clustering de usuarios para ofrecer una propuesta de valor al usuario..
3. **Desarrollo de la API**. Ésta, tras una llamada de la aplicación web, devuelve al usuario datos de sus transacciones y recomendaciones que encajan con su perfil.
4. **Desarrollo de la aplicación web**. Encargada de visualizar datos de usuarios obtenidos a través de la API.

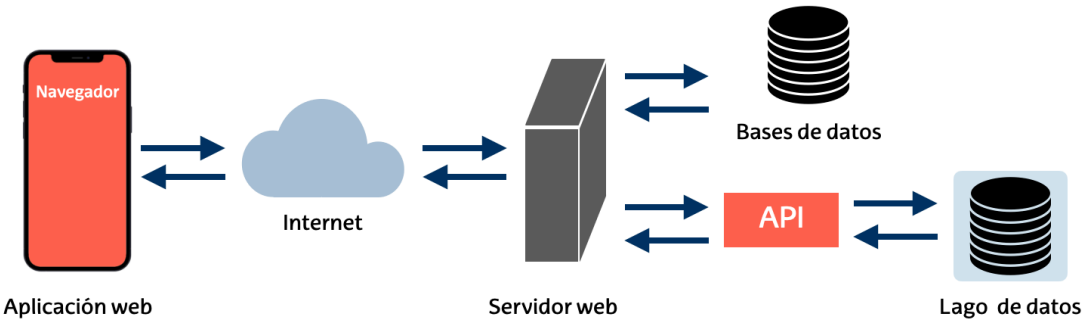


Figura 14: Esquema general de la solución del proyecto.

El orden de desarrollo de los bloques funcionales, disponible en la Descripción de tareas, se ha definido así ya que unos son dependientes de otros. Por ejemplo, para poder realizar el clustering de los usuarios es necesario que la información se encuentre disponible y ordenada, por lo que la creación del lago de datos ha de ser anterior a el análisis.

7.1. Data Lake

El dataset utilizado para este proyecto es un ejemplo disponible en MongoDB para pruebas de desarrollo de forma gratuita. Este dataset [21] se encuentra disponible en una base de datos que a su vez se divide en colecciones.

Una colección de MongoDB es equivalente al concepto de tabla de base de datos. Como se puede observar en la Figura 15, una tabla almacena registros o filas, mientras que una colección almacena documentos. Este es el punto principal que separa a una base de datos SQL y una NoSQL. Los registros de una base de datos están compuestos por diferentes columnas y esas son las mismas para todos los registros de una misma tabla. Esto no sucede en los documentos.

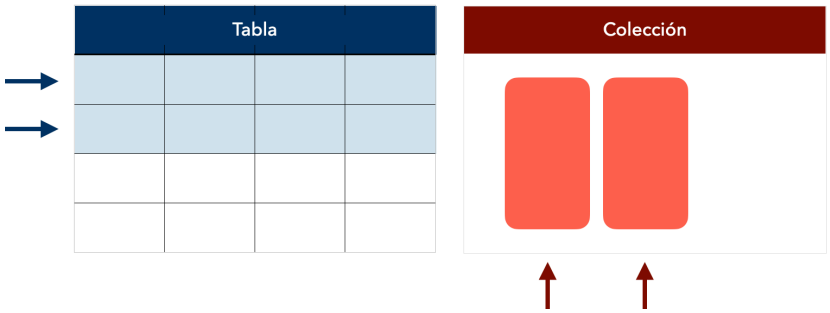


Figura 15: Comparativa de una Tabla SQL y una colección NoSQL.

Un documento está formado por claves o keys y dentro de una misma colección puede haber documentos con variaciones en esas claves. En NoSQL es conocido como Schema Free, se puede visualizar un ejemplo en la Figura 16. En la Figura se muestra un ejemplo de documento, pero en esa misma colección puede existir otro documento que no disponga de la key "username" o nombre de usuario.

Figura 16: Ejemplo de documento del dataset utilizado.

Este concepto se puede presentar como una gran ventaja, ya que aporta una gran flexibilidad. Pero por otro lado, puede resultar muy desordenado. Es imprescindible plantear las necesidades de almacenamiento requeridas para cada dato.

Trabajar con diferentes tipos de bases de datos es una buena alternativa debido a que cada una de ellas aporta ventajas en función del objetivo de la misma. Con una clara necesidad de poder tratar con diferentes bases de datos y formas de almacenar esa información, surge la necesidad de uso de un Data Lake. Este concepto se introduce en lel Contexto, en la sección 2.1.

MongoDB Atlas Data Lake, lanzado en 2019, se creó para llevar el modelo de documento flexible, el lenguaje de consulta y las herramientas de MongoDB al dominio de Data Lakes. Atlas Data Lake es un motor de procesamiento de consultas bajo demanda de múltiples inquilinos que le permite utilizar el lenguaje de consulta MongoDB, MQL, en los datos del almacén de objetos en la nube en múltiples formatos. Algunos de los formatos incluidos son JSON, BSON, CSV, Avro, Parquet, etc. Para todo esto, Atlas Data Lake implementa múltiples nodos de cómputo representados en la Figura 17.

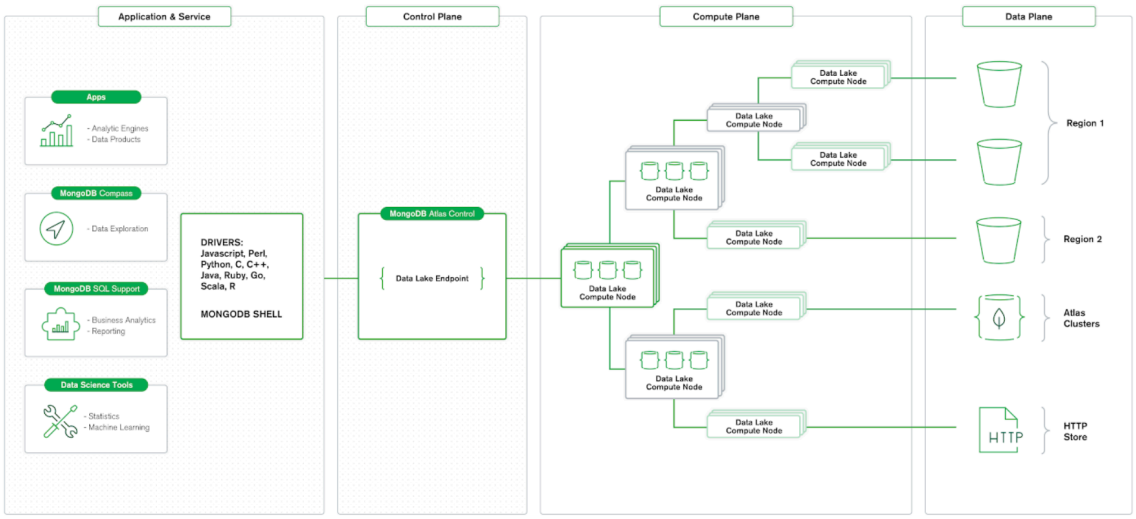


Figura 17: Esquema general del Data Lake de MongoDB.

La configuración y el enlace de las bases de datos con el Data Lake se puede realizar mediante modificaciones en los archivos de configuración o haciendo uso de una cómoda interfaz gráfica disponible en la propia web de MongoDB. Ambos casos se muestran en la Figura 18.

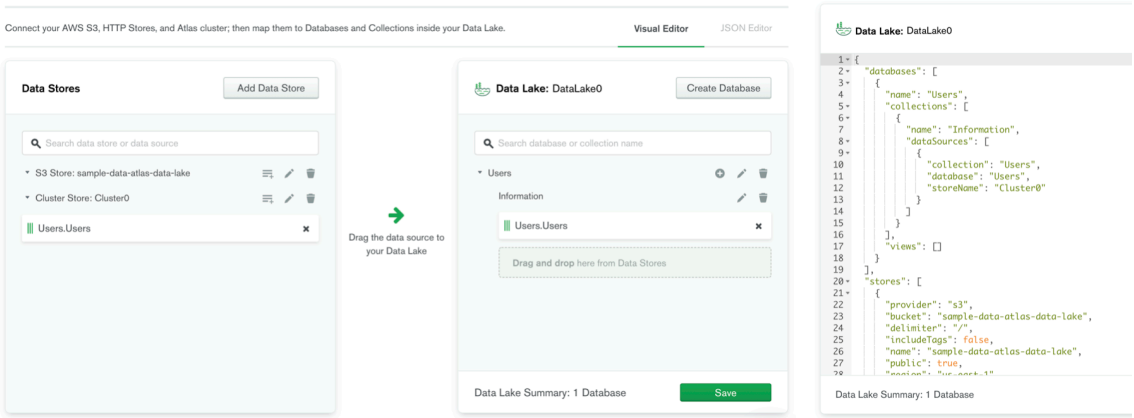


Figura 18: Métodos de documentación del Data Lake.

En el proyecto se trabaja con bases de datos basadas en documentos. A largo plazo se plantea usar diferentes inquilinos con diferentes formas de almacenar la información, como Amazon o Azure. Esto sucede ya que se trata de una aproximación inicial de la arquitectura de la empresa y se trata de los primeros servicios ofrecidos.

El proyecto dispone de tres bloques de almacenamiento principales, los cuales se pueden visualizar en la Figura 19:

1. **Banco**. Base de datos formada por los datos de usuario y sus cuentas junto a las transacciones.
2. **Analisis**. Base de datos con la información obtenida en el análisis, es decir, la propuesta de valor.
3. **Aplicación**. Base de datos que relaciona al usuario registrado en la app con un token representativo de la base de datos de análisis.

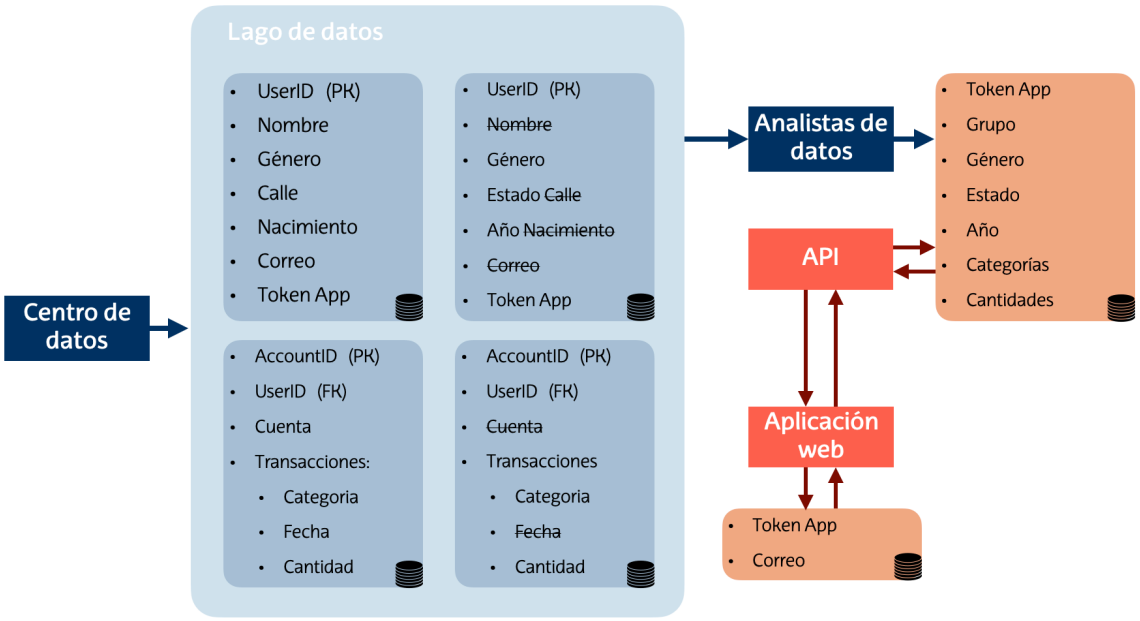


Figura 19: Estructura de los diferentes bases de datos.

Tal y como se menciona en el Contexto 2.1, el gobierno de datos es necesario para disponer de un contenido de calidad. Una vez escogido el dataset se realiza una limpieza del mismo de forma que los datos se encuentran correctamente formateados siguiendo unos namings definidos.

```
gender: "female"
address: "9286 Bethany Glens
Vasqueztown, CO 22939"
birthdate: 1977-03-02T03:20:31.000+00:00
```

Figura 20: Forma de definir la dirección de usuario.

Un ejemplo de esto es la forma de definir las direcciones. En la Figura 20 se muestra el caso de un usuario. El naming "address" o dirección del usuario se define como "Calle, Abreviatura estado Código Postal". Otro ejemplo es la fecha, también disponible en la figura anterior. Para esta se ha escogido el formato Date de MongoDB.

7.1.1. Base de datos del banco

Este bloque se divide en dos documentos diferenciados:

- **Información personal**. Documento con información relativa al cliente como fecha de nacimiento, correo, etc, disponible en la Figura 21.
- **Información de cuenta**. Documento es relativo a la cuenta con el saldo disponible, transacciones, etc. Se relaciona con el documento anterior añadiendo el id de usuario. Este documento se encuentra disponible en la Figura 22.


```
_id: ObjectId("60e49e5d11b9d2175e29fbc3")
name: "Elizabeth Ray"
gender: "female"
address: "9286 Bethany Glens
Vasqueztown, CO 22939"
birthdate: 1977-03-02T03:20:31.000+00:00
email: "alicia.fl@outlook.com"
token: "a6c84f3d50552068b2c3dd3972216aec4dc3ce39deeb09524522a6bf1e83cf55"
```


Figura 21: Base de datos de la información personal de los usuarios.


```
_id: ObjectId("60e49e5e11b9d2175e29fbc4")
user_id: ObjectId("60e49e5d11b9d2175e29fbc3")
account: 371138
transactions: Array
  0: Object
    date: 2013-10-18T02:00:00.000+00:00
    category: "Leisure"
    price: "20.01"
  1: Object
  2: Object
  3: Object
  4: Object
  5: Object
  6: Object
  7: Object
  8: Object
  9: Object
  10: Object
  11: Object
  12: Object
  13: Object
token: "a6c84f3d50552068b2c3dd3972216aec4dc3ce39deeb09524522a6bf1e83cf55"
balance: "8200"
```


Figura 22: Base de datos de la información de cuenta de los usuarios.

Existen dos puntos interesantes a resaltar. Por un lado, en todos los documentos se encuentra disponible una key "Token App" que identifica al usuario de forma anónima. Este token es aquel que se utiliza para conectar al usuario de la aplicación con sus respectivos documentos.

Por otro lado, en este bloque se encuentran los documentos replicados. Esto se hace para que los analistas puedan acceder a la información de estudio sin conocer ciertos datos personales del cliente y no comprometer así su privadidad.

```
_id: ObjectId("60e49e5d11b9d2175e29fbc3")
name: "e59d8e9b-db5f-4b8d-b897-cb00bcecdaf7"
gender: "female"
address: "********* CO"
birthdate: "1977 *****"
token: "a6c84f3d50552068b2c3dd3972216aec4dc3ce39deeb09524522a6bf1e83cf55"
```


Figura 23: Base de datos replicadas de la información personal de los usuarios.


```
_id: ObjectId("60e49e5e11b9d2175e29fbc4")
user_id: "60e49e5d11b9d2175e29fbc3"
account: "ed295f38-3b02-4e69-ac30-eef86a06187c"
transactions: Array
  0: Object
      date: "2013"
      category: "Leisure"
      price: "20.01"
  1: Object
  2: Object
  3: Object
  4: Object
  5: Object
  6: Object
  7: Object
  8: Object
  9: Object
  10: Object
  11: Object
  12: Object
  13: Object
token: "a6c84f3d50552068b2c3dd3972216aec4dc3ce39deeb09524522a6bf1e83cf55"
balance: 8200
```


Figura 24: Base de datos replicadas de la información de cuenta de los usuarios.

La anonimización de datos es el proceso de cifrar o eliminar información de identificación personal de los conjuntos de datos. En términos empresariales, uno de los principales motivos por los que se usa es para proteger datos previamente clasificados como sensibles o identificadores.

En la base de datos se ha usado:

- Enmascaramiento. Técnica de anonimización que permite ocultar una parte importante de los datos con caracteres aleatorios u otros datos. Esta metodología se usa en la dirección o la fecha de nacimiento.
- Tokenización. Técnica que sustituye un elemento de datos sensibles con un equivalente no sensible, denominado token, que no tiene un significado o valor extrínseco o explotable. Esto se usa en el nombre de usuario.

En la Figura 25 se muestran los documentos anteriormente mencionados de la base de datos del banco.

Bank

DATABASE SIZE: 11.89MB INDEX SIZE: 256KB TOTAL COLLECTIONS: 4

CREATE COLLECTION

Collection Name	Documents	Documents Size	Documents Avg	Indexes	Index Size	Index Avg
Accounts	1746	5.51MB	3.23KB	1	64KB	64KB
Accounts_r1t	1746	5.62MB	3.3KB	1	60KB	60KB
Users	1746	411.69KB	242B	1	64KB	64KB
Users_r1t	1746	366.54KB	215B	1	68KB	68KB

Figura 25: Documentos de la colección del banco.

7.1.2. Base de datos del análisis

En esta base de datos se incluye la información relativa al análisis. Por un lado, un resumen de los datos de gastos de clientes por categorías en un array. Por otro lado, también existe una key "profile" o perfil que reconoce en que grupo se encuentra el usuario. Al igual que en los documentos anteriores, se identifica al cliente con un token de usuario.

```
_id: ObjectId("613a27bba1bf2dc7542c1ea2")
user_id: "60e49e5d11b9d2175e29fbc3"
account: 371138
token: "a6c84f3d50552068b2c3dd3972216aec4dc3ce39deeb09524522a6bf1e83cf55"
profile: 6
expenses: Array
  0: 18.89
  1: 61.72
  2: 32.79
  3: 232.49
  4: 20.01
```

Figura 26: Base de datos de la información de análisis.

7.1.3. Base de datos de la aplicación

Este documento es el más simple de los tres. Cuando el usuario se conecta a la aplicación lo hace gracias al mail de usuario. Este documento, disponible en la Figura 27, recoge el mail de usuario y el token. Cuando un usuario inicia sesión a través de Github busca el token en la base de datos, para conocer su token y poder realizar consultas al resto de las bases de datos.

```
_id: ObjectId("60e49e5d11b9d2175e29fbc3")
name: "Elizabeth Ray"
email: "alicia.fl@outlook.com"
token: "a6c84f3d50552068b2c3dd3972216aec4dc3ce39deeb09524522a6bf1e83cf55"
```

Figura 27: Base de datos de la información de la aplicación.

Una vez ordenadas las bases de datos y sus respectivas colecciones, además de organización y definición de namings, el siguiente paso es el análisis con Machine Learning. Es importante resaltar que la base de datos de análisis se completa una vez realizado el clustering. En este punto se define la forma de las colecciones y los datos necesarios para ofrecer el servicio al cliente.

7.2. Clustering

En el Contexto, en la sección 2.2.2, se definen los cinco pasos a seguir a la hora de realizar clustering a una población. Para desarrollar este apartado de la solución se sigue ese mismo esquema de cara a dejar claro el desarrollo de este punto.

Esos pasos son:

1. Selección de datos.
2. Definición de una medida de proximidad o criterio de distancia.
3. Selección del tipo de algoritmo de clustering.
4. Abstracción de los datos.
5. Validación de los resultados.

7.2.1. Selección de datos

La selección de los datos para realizar el análisis comienza en el punto anterior, en el cual se definen los datos que se han tomado y como se han organizado.

De cara a la realización del clustering se han tenido en cuenta los siguientes datos:

- Estado
- Fecha de nacimiento
- Transacciones
    - Compras
    - Supermercado
    - Transporte
    - Restaurantes
    - Ocio

Algunos, como el género, no se han tenido en cuenta debido a que no existe un patrón fijo y distorsionan los resultados. Cuando exista una mayor cantidad de usuarios y se definan más concretamente sus perfiles, es posible que alguno de los datos existentes no utilizados en esta iteración sean considerados para el clustering.

7.2.2. Definición de un criterio de distancia

Los criterios de distancia se escogen junto a la selección del algoritmo en el siguiente punto. Es importante conocer que se tratan de datos cuantitativos. Esto conlleva a que, tal y como se explica en el Contexto 2.2.2, se encuentra limitado a ciertos tipos de distancia.

7.2.3. Selección del tipo de algoritmo

Dentro de las características analizadas en el Contexto en la sección 2.2.1, la considerada más relevante es el número de clusters. Para hacer una analítica más completa de cara a la implantación del algoritmo en el proyecto, se ha considerado escoger uno de cada subtipo. De esta forma, se realiza un estudio de como se comportan y de que metodología devuelve mejores resultados.

Haciendo uso de la división natural realizada en el estado del arte, se completa el listado con ejemplos de algoritmos existentes en la librería Scikit Learn escogida anteriormente. Los métodos resaltados en negrita son aquellos que se van a utilizar, por lo que posteriormente se definen concretamente.

- **Según si se conoce nºclusters:**
  - Jerárquico
    - Aglomerativos: **Agglomerative**
  - Particional.
    - Agrupación exclusiva.
      - De partición del espacio: **Kmeans**
      - Basada en densidades: **DBSCAN**

En la Figura 28 se pueden observar los algoritmos escogidos y sus características principales. Es importante destacar que todos ellos son para una gran cantidad de muestras de usuarios. Aunque la base de datos actual no dispone de una gran escala, se prevee que sea necesario disponer de un algoritmo adaptado al caso.

Method name	Parameters	Scalability	Usecase	Geometry (metric used)
K-Means	number of clusters	Very large `n_samples`, medium `n_clusters` with MiniBatch code	General-purpose, even cluster size, flat geometry, not too many clusters, inductive	Distances between points
Affinity propagation	damping, sample preference	Not scalable with n_samples	Many clusters, uneven cluster size, non-flat geometry, inductive	Graph distance (e.g. nearest-neighbor graph)
Mean-shift	bandwidth	Not scalable with `n_samples`	Many clusters, uneven cluster size, non-flat geometry, inductive	Distances between points
Spectral clustering	number of clusters	Medium `n_samples`, small `n_clusters`	Few clusters, even cluster size, non-flat geometry, transductive	Graph distance (e.g. nearest-neighbor graph)
Ward hierarchical clustering	number of clusters or distance threshold	Large `n_samples` and `n_clusters`	Many clusters, possibly connectivity constraints, transductive	Distances between points
Agglomerative clustering	number of clusters or distance threshold, linkage type, distance	Large `n_samples` and `n_clusters`	Many clusters, possibly connectivity constraints, non Euclidean distances, transductive	Any pairwise distance
DBSCAN	neighborhood size	Very large `n_samples`, medium `n_clusters`	Non-flat geometry, uneven cluster sizes, transductive	Distances between nearest points
OPTICS	minimum cluster membership	Very large `n_samples`, large `n_clusters`	Non-flat geometry, uneven cluster sizes, variable cluster density, transductive	Distances between points
Gaussian mixtures	many	Not scalable	Flat geometry, good for density estimation, inductive	Mahalanobis distances to centers
BIRCH	branching factor, threshold, optional global clusterer.	Large `n_clusters` and `n_samples`	Large dataset, outlier removal, data reduction, inductive	Euclidean distance between points

Figura 28: Selección de algoritmos disponibles en Scikit Learn y sus características.

A continuación se definen los diferentes tipos de algoritmos escogidos, para escoger el que mejores resultados ofrece. El análisis de los resultados se realiza en el apartado dedicado a ello.

7.2.3.1. Aglomerativo

Para comprender este punto es necesario definir una serie de conceptos antes de desarrollar la solución.

En primer lugar, el algoritmo aglomerativo sigue un enfoque de abajo hacia arriba. Tal y como se explica en el Contexto 2.2.1, comienza con muchos grupos pequeños y los combina para crear grupos más grandes hasta que sólo queda un cluster.

En segundo lugar, el criterio que define como crear esas agrupaciones es la distancia euclídea. La distancia euclídea entre dos puntos p y q es la longitud del segmento que une ambos puntos. En coordenadas cartesianas, la distancia euclídea se calcula empleando el teorema de Pitágoras. Si se trata de un espacio bidimensiones en el que cada punto está definido por las coordenadas (x,y), la distancia euclídea viene dada por la ecuación:

$$d_{euc}(p,q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2} \tag{7.1}$$

La versión generalizada para un espacio n-dimensional donde cada punto está definido por un vector de n coordenadas: p=(p1,p2,p3,...,pn) y q=(q1,q2,q3,...,qn).

$$d_{euc}(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{7.2}$$

Por último, el criterio de enlace se refiere a cómo se calcula la distancia entre clusters. En la Figura 29 se muestran todos los tipos que existen en la librería. El criterio escogido es el Ward, el cual es la suma de las diferencias al cuadrado dentro de todos los clusters.

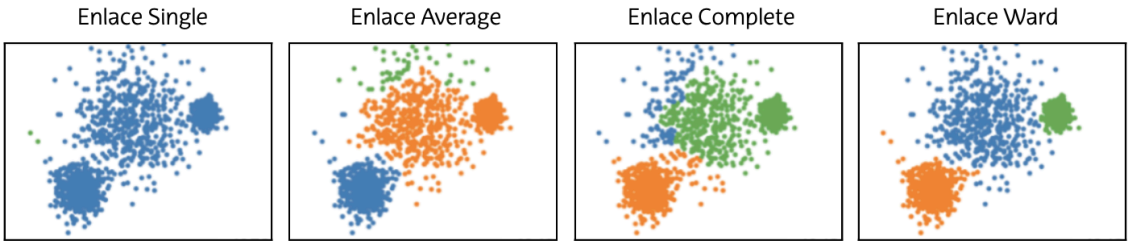


Figura 29: Muestras de funcionamiento de criterio de enlace.

Para explicar el funcionamiento del código desarrollado se utiliza el Pseudocódigo 30.

```
   /* 1. Preparación de la información                    */
1  Dataframe = normalizar(Dataframe)
2  X = Datos escogidos
   /* 2. Visualización del dendograma                     */
3  Dendograma = dendograma(X)
4  Visualización del dendograma
   /* 3. Clustering aglomerativo                          */
5  Modelo = clustering Aglometarivo(clusters,método)
6  Fit del Modelo(X)
7  Etiquetas = etiquetas del Modelo
```

Figura 30: Pseudocódigo del algoritmo.

Los pasos definidos en el pseudocódigo son los siguientes:

1. Se normaliza la información de forma que todos los valores quedan en una escala común y se escogen los datos relevantes para el uso en el algoritmo.
2. Se visualiza un dendograma que muestra la distancia euclídea para cada grupo.
3. Se realiza el clustering definiendo como criterio de parada al número de clusters escogido tras el análisis del dendograma.

En la Figura 31 se nuestra el resultado de la ejecución del segundo paso. Se puede observar como la distancia entre todos los usuarios es muy pequeña. En el eje Y se observa que para para 10 clusters la distancia es menor de dos.

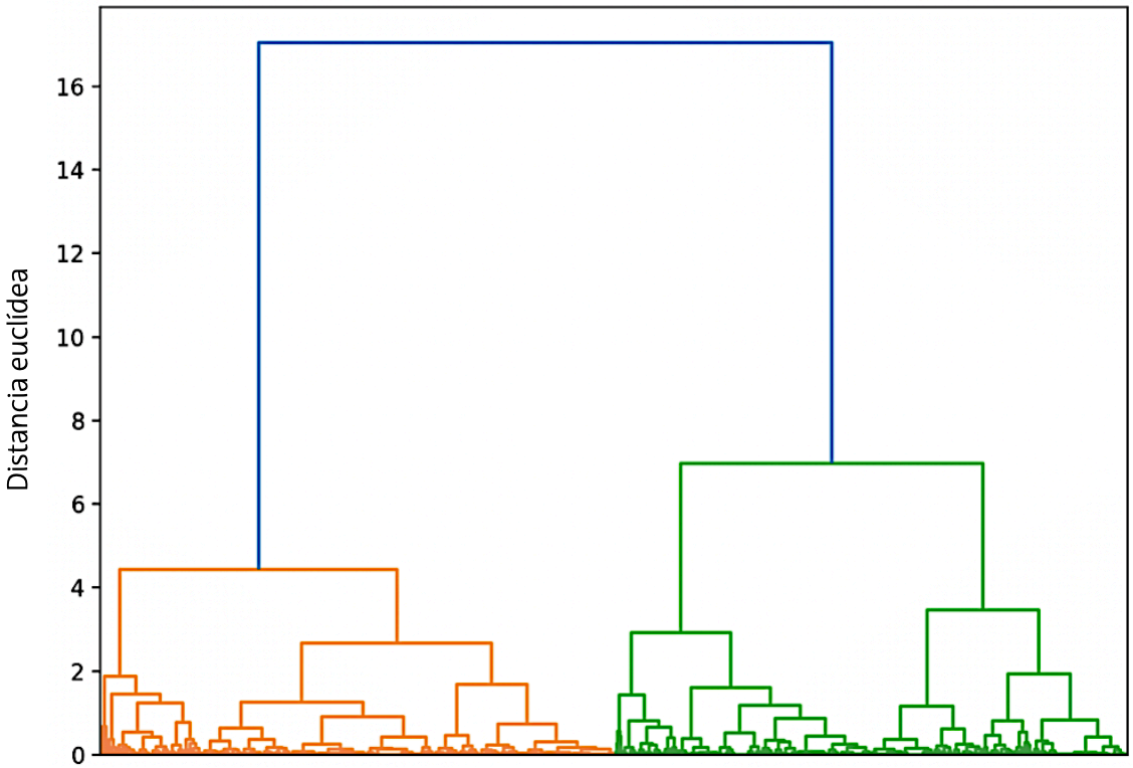


Figura 31: Dendograma obtenido con el algoritmo aglomerativo.

7.2.3.2. Kmeans

Al igual que en el algoritmo anterior, es necesario conocer una serie de conceptos antes de conocer el desarrollo realizado.

Por un lado, el algoritmo de clustering Kmeans se utiliza para encontrar grupos que no se han etiquetado explícitamente en los datos. Al tratarse de un algoritmo particional es necesario conocer el número final de clusters. Para ello existen diferentes métodos que den un aproximación inicial. Por un lado, el método del codo y por otro lado utilizar un algoritmo jerárquico para estimar ese valor.

Por otro lado, la inercia es la suma de las distancias al cuadrado de cada objeto del cluster a su centroide. La inercia es la métrica que se utiliza para definir realizar el método del Codo previo a la ejecución de Kmeans. Cuando los valores de inercia son altos el resultado no es bueno. En la fórmula k representa la cantidad de grupos y μ representa el centroide de cada grupo.

$$argminS \sum_{i=1}^{k} \sum_{x \epsilon S_i} ||x - \mu_i||^2 \tag{7.3}$$

El último concepto que es necesario conocer para comprender el desarrollo es los pasos que sigue el algoritmo de Kmeans para obtener los clusters son los siguientes:

1. Se determina el valor "k" o número de clusters.
2. Se selecciona de forma aleatoria k centroides distintos. Este número de centroides vienen definidos por el número de clusters, es decir, si existen "k" clusters se definen "k" centroides.
3. Se mide la distancia, en este caso la euclídea definida en el apartado anterior, entre cada punto y el centroide.
4. Se asigna cada punto al grupo más cercano.
5. Se recoloca el centroide de cada cluster.
6. Se reasigna cada punto a un nuevo centroide más cercano.

Los dos últimos puntos se repiten continuamente hasta que converge la solución y se obtiene la solución final.

Para explicar el funcionamiento del código desarrollado se utiliza el Pseudocódigo 32.

```
   /* 1. Preparación de la información                    */
1  Dataframe = normalizar(Dataframe)
2  X = Datos escogidos
   /* 2. Método del codo                                  */
3  for _ in range(i) do
4  |  Modelo = Kmeans(i)
5  |  Fit Predict del Modelo(X)
6  end
   /* 3. Clustering kmeans                                */
7  Modelo = Kmeans(clusters)
8  Etiquetas = Fit Predict del Modelo(X)
```

Figura 32: Pseudocódigo del algoritmo.

En la Figura 33 se ve el resultado del método del codo. Se puede ver que después de 2 no hay una disminución significativa en WCSS, por lo que los valores 2 y 3 son los mejores aquí. Resulta aconsejable escoger el número donde se forma el codo. Existen muchas ocasiones en las que el gráfico no es tan intuitivo, pero con la práctica se vuelve más fácil. Como los puntos están muy juntos tarda muy poco en converger a un buen valor de inercia.

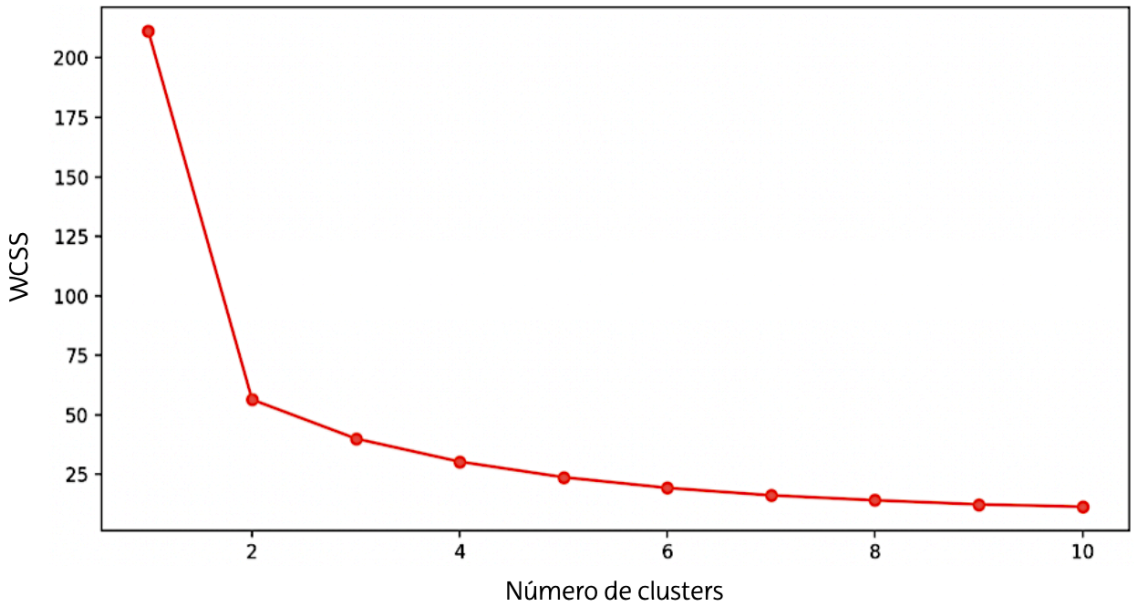


Figura 33: Solución del método del codo.

7.2.3.3. Dbscan

Una vez realizado el estudio de Kmeans, es necesario comprender la diferencia que éste presenta respecto al algoritmo basado en densidades DBSCAN. El algoritmo Kmeans es bueno encontrando agrupaciones con forma esférica o convexa con poco ruido, pero falla cuando se trata de formas arbitrarias. Tal y como se observa en la Figura 34, DBSCAN soluciona ese problema definiendo un mínimo de observaciones vecinas dentro de un radio de proximidad. Ese radio de proximidad es el valor epsilon.

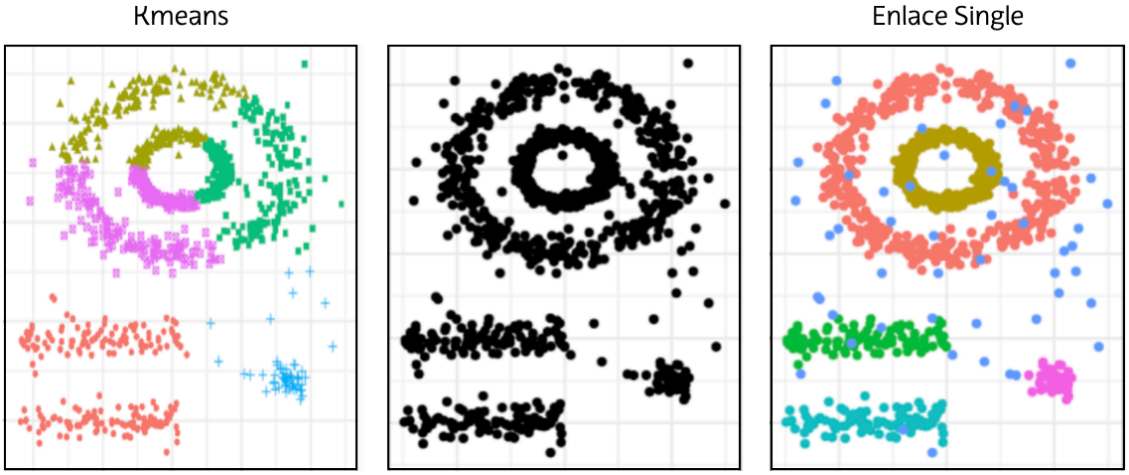


Figura 34: Comparación gráfica del algoritmo Kmeans y Dbscan.

Dbscan devuelve ciertos puntos como ruido con valor -1. Estos corresponden con los usuarios en azul de la Figura 34. La cantidad de ruido es un valor muy relevante a la hora de escoger la solución más adecuada.

```
   /* 1. Preparación de la información                    */
1  Dataframe = normalizar(Dataframe)
2  X = Datos escogidos
   /* 2. Método del codo                                  */
3  metodo del codo()
   /* 3. DBSCAN                                           */
4  Modelo = DBSCAN(epsilon).fit(X)
5  Fit del Modelo (X)
6  Etiquetas = etiquetas del Modelo
7  ruido = Etiquetas con valor -1
```

Figura 35: Pseudocódigo del algoritmo.

Los pasos seguidos se encuentran en la Figura 35. Primero se normaliza la información y se prepara la información al igual que en el resto de los casos. En segundo lugar se aplica el método del codo. El desarrollo del método del codo no es el mismo que en el caso de Kmeans. Por último, se aplica Dbscan en función del valor de epsilon adecuado.

En la Figura 36 se muestra el método del codo. Esta técnica consiste en fijar un valor minPts y, partir de ahí, crear un gráfico con todas los radios eps. Cuando los radios comienzan a aumentar de forma exponencial significa que se aleja de la zona de alta densidad y se entra en la zona de baja densidad.

Un valor de eps = 0,01 es mucho más restrictivo, admitiendo solo los puntos de zonas muy densas. Por otro lado, un valor de eps = 0,02 es más relajado con valores de zonas menos densas.

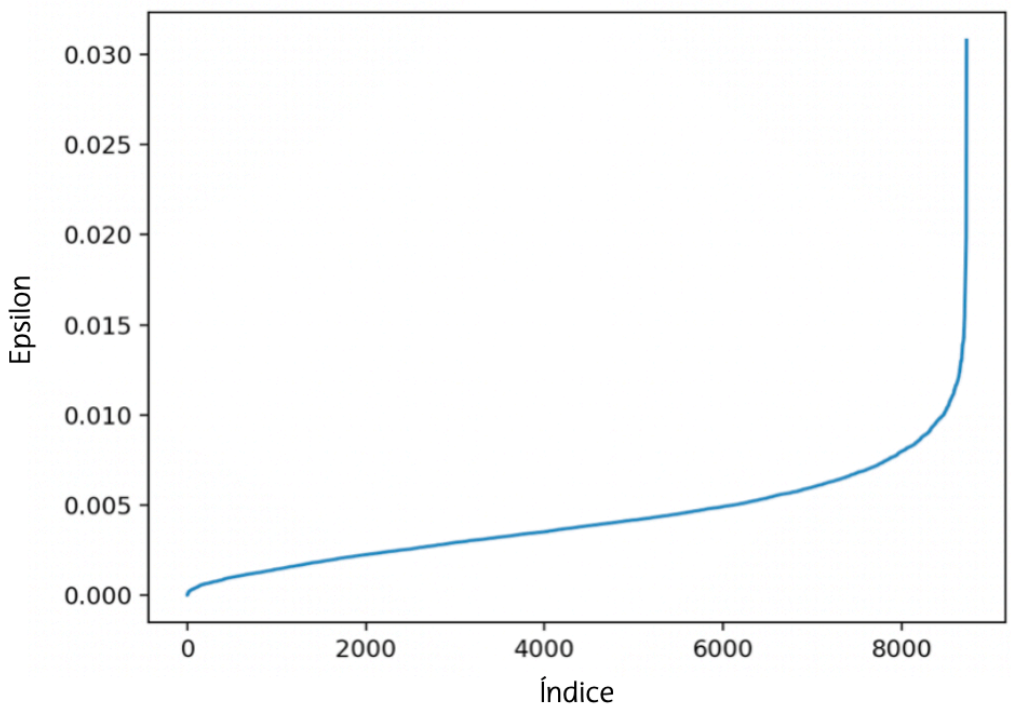


Figura 36: Método del codo para DBSCAN.

El algoritmo DBSCAN necesita dos parámetros:

- Epsilon, ϵ. Radio que define la región vecina a una observación.
- Minimum points, minPts. Número mínimo de puntos que, dado un valor epsilon, tiene que haber para que se considere que forman un clúster

Cada observación del set de datos se puede clasificar en una de las categorías:

- Core point. Observación que tiene en su radio epsilon un número de observaciones vecinas igual o mayor a minPts.
- Border point. Observación no satisface el mínimo de observaciones vecinas para ser Core point como en el caso anterior. En cambio, pertenece al radio epsilon de otra observación que sí lo es.
- Noise u outlier. Observación que no esninguno de los casos anteriores.

Por último, se pueden definir tres niveles de conectividad entre observaciones:

- Directamente alcanzable . A es directamente alcanzable desde B si A forma parte del radio epsilon de B y B es un core point.
- Alcanzable. A es alcanzable desde B si existe una secuencia de core points que van desde B a A.

- Densamente conectadas. A y B están densamente conectadas si existe una observación core point C tal que A y B son alcanzables desde C.

La Figura 37 muestra un ejemplo de observaciones y sus respectivas conexiones. El valor definido para minPts es 4. A y el resto de observaciones marcadas en rojo son core points, ya que todas ellas contienen al menos 4 observaciones vecinas, ellas incluidas, en su región epsilon. Como todas ellas son alcanzables entre si, forman un cluster. B y C no son core points, ya que no poseen 4 observaciones vecinas, pero si son alcanzables desde A, por lo que pertenecen al mismo cluster. En cambio, la observación D no es core point y además no es directamente alcanzable, por lo que se considera la categoría restante o ruido.

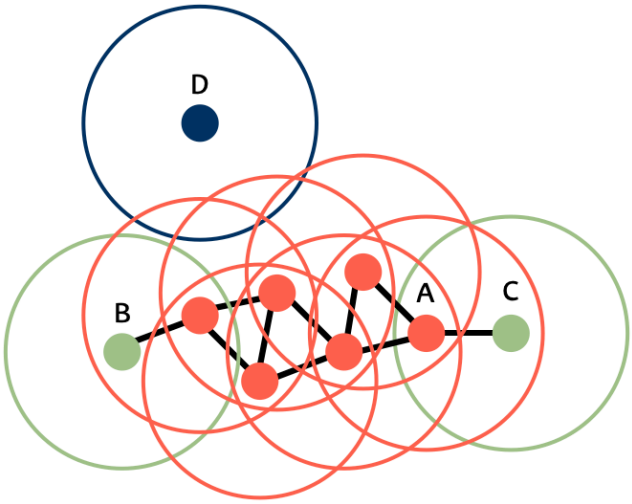


Figura 37: Método de Dbscan.

Una vez comprendidos los conceptos del funcionamiento de Dbscan, se siguen los siguientes pasos:

1. Para cada observación x_i calcular la distancia entre ella y el resto. Si en su región epsilon hay un número de observaciones mayor o igual que minPts, se marca como core point, de lo contrario se marca como visitada.
2. Para cada observación xi marcada como core point, si no ha sido asignada a ningún cluster, se crea uno nuevo y se asigna la obervación a él. Encontrar recursivamente todas las observaciones densamente conectadas a ella y asignarlas al mismo cluster.
3. Se itera el mismo proceso para todas las observaciones no marcadas como visitadas.
4. Las observaciones que se han marcado como visitadas, pero no pertenecen a ningún cluster, se marcan como ruido.

El resultado de esto es que todos los clusters cumplen dos propiedades. La primera es que los puntos de un mismo cluster están densamente conectados entre ellos. La segunda es que si una observación es densamente alcanzable desde cualquier otra observación de un cluster, entonces esa observación también pertenece al cluster.

7.2.4. Abstracción de los datos

Se toma una pequeña muestra de los usuarios de forma aleatoria y se asigna clusters en función de sus características haciendo un estudio de los valores existentes. Por ejemplo, para definir al usuario respecto a su edad se observa el rango de edades de la base de datos y se agrupa los que se encuentran en rangos similares. Repitiendo esto con todas las características, se obtiene una solución "ideal".

Esta solución ideal obtenida es la que se utiliza en las funciones de evaluación de los algoritmos, esto se puede observar en la figura 38 y **??**. A las funciones de evaluación se les pasa el parámetro clusters obtenidos mediante los algoritmos y clusters de la solución "ideal".

$$labels\_true = [0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1] \tag{7.4}$$

$$labels\_pred = [0, 0, 0, 1, 1, 2, 3, 4, 4, 4, 5] \tag{7.5}$$

Parameters:

labels_true : ***int array, shape = [n_samples]***
A clustering of the data into disjoint subsets.

labels_pred : ***int array-like of shape (n_samples,)***
A clustering of the data into disjoint subsets.

Figura 38: Parámetros para la función de validación.

7.2.5. Validación de los resultados

Para la validación de los resultados se escogen una serie de métricas y se visualizan los resultados. Una vez realizados los pasos anteriores, se escoge el métodos con mejores resultados y el número de clusters correspondiente.

7.2.5.1. Definición de métricas

Las funciones escogidas para evaluar el comportamiento del algoritmo para este caso son:

- NMI: La información mutua normalizada es una normalización de la puntuación de información mutua con resultados escalados entre 0 y 1. Cuando la correlación es perfecta el resultado es 1, mientras que si no existe información mutua se obtiene un 0. Es simétrico, se obtiene el mismo valor si se varían los labels de la Figura **??**.

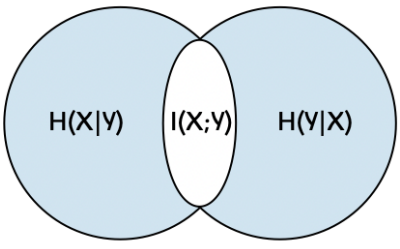


Figura 39: Gráfico de información mutua.

- AMI: La información mutua ajustada es un ajuste de la puntuación de la anterior para tener en cuenta el azar. Explica el hecho de que la información mutua es generalmente más alto para dos clusters generados con una mayor cantidad de agrupaciones, independientemente de si en realidad se comparte más información. Al igual que en la función anterior es simétrico, se obtiene el mismo valor si se varían los labels.

$$AMI(U,V) = \frac{MI(U,V) - E\{MI(U,V)\}}{max\{H(U),H(V)\} - E\{MI(U,V)\}} \tag{7.6}$$

- Homogeneidad completa: V-Measure es idéntico a NMI con el método de promedio aritmético. Los dos análisis no son simétricos, uno se convierte en el otro si cambias los labels.
    - Homogeneidad: los grupos contienen sólo miembros de una misma clase.
    - Completa: los miembros de una clase están en el mismo cluster.

Las métricas han sido escogidas para complementar una a la otra. La forma de analizar la solución es diferente una de la otra, por lo que al realizar el análisis basándose en tres criterios resulta más completa.

7.2.5.2. Análisis de gráficas

Una vez definidas las métricas a utilizar para evaluar el comportamiento de los diferentes algoritmos, es momento de mostrar los resultados y evaluarlos.

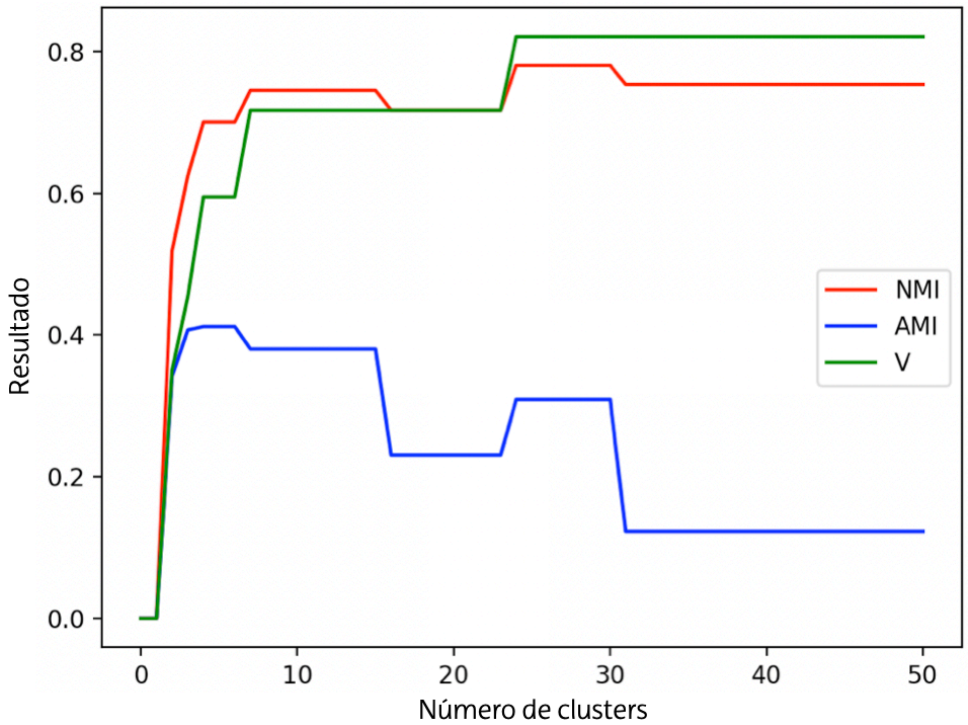


Figura 40: Resultado de algoritmo aglomerativo.

En primer lugar, en la Figura 40 se muestra el resultado obtenido para el algoritmo Aglomerativo. Para las métricas NMI y V se obtienen resultados similares al método anterior. En cambio, en AMI se observan mejores resultados. Para un valor de número de clusters de 10 aproximadamente se mantiene el valor de 0.4. Este método devuelve mejores resultados que el anterior.

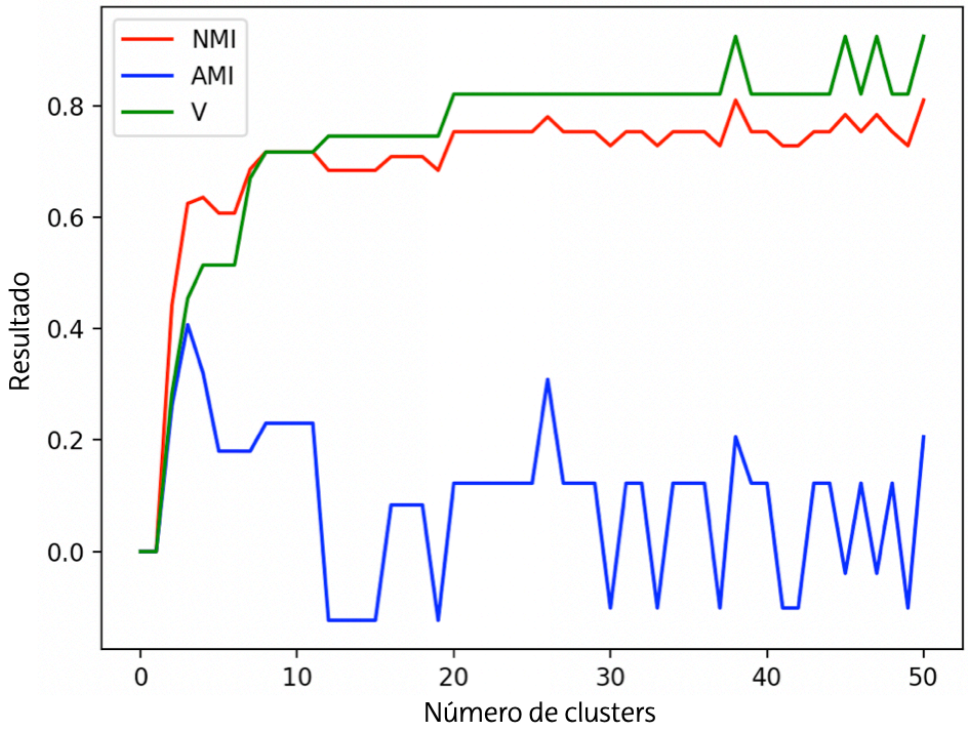


Figura 41: Resultado de algoritmo Kmeans.

En segundo lugar, en la Figura 41 se muestra el resultado obtenido para el algoritmo Kmeans. Se puede observar que las métricas NMI y V dan buenos resultados para todo el espectro de número de clusters. En cambio, AMI, tiene el punto más alto en 0.4. Este es un resultado medio que luego empeora. Para los 10 cluster se obtiene 0.2.

El último método a analizar es DBSAN. Este es el más complejo de analizar, por lo que se han dividido los resultados en dos figuras para simplificarlo.

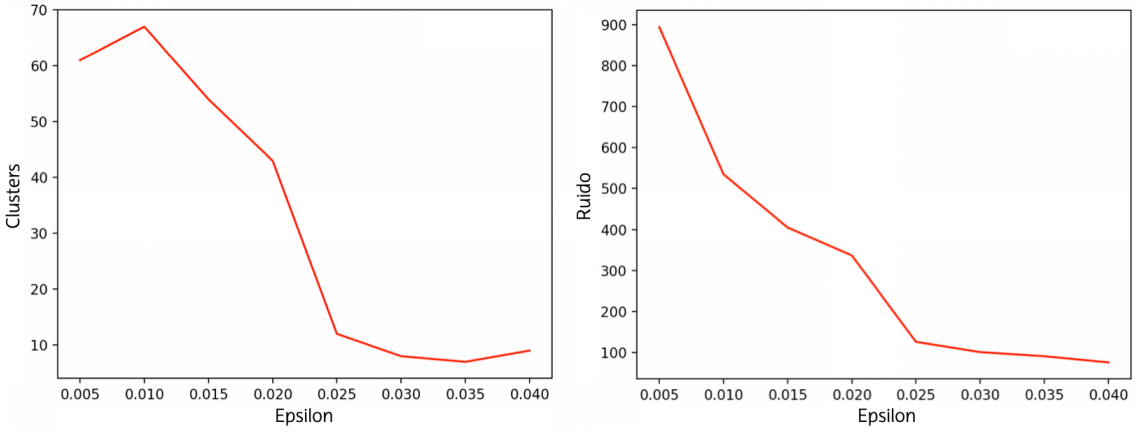


Figura 42: Parámetros del resultado de dbscan.

En la Figura 42 en la gráfica de la izquierda se observa como existe una relación entre el valor de epsilon, explicado en la sección de Selección del algoritmo de clustering, y el número de clusters. Esta gráfica es la que sirve de apoyo para el análisis de resultados. Por otro lado, en la gráfica de la derecha se observa una relación entre el valor de epsilon y el número de usuarios ruido que no encajan en ningún grupo. Se puede observar que, a medida que aumenta el valor de epsilon, disminuye la cantidad de ruido existente.

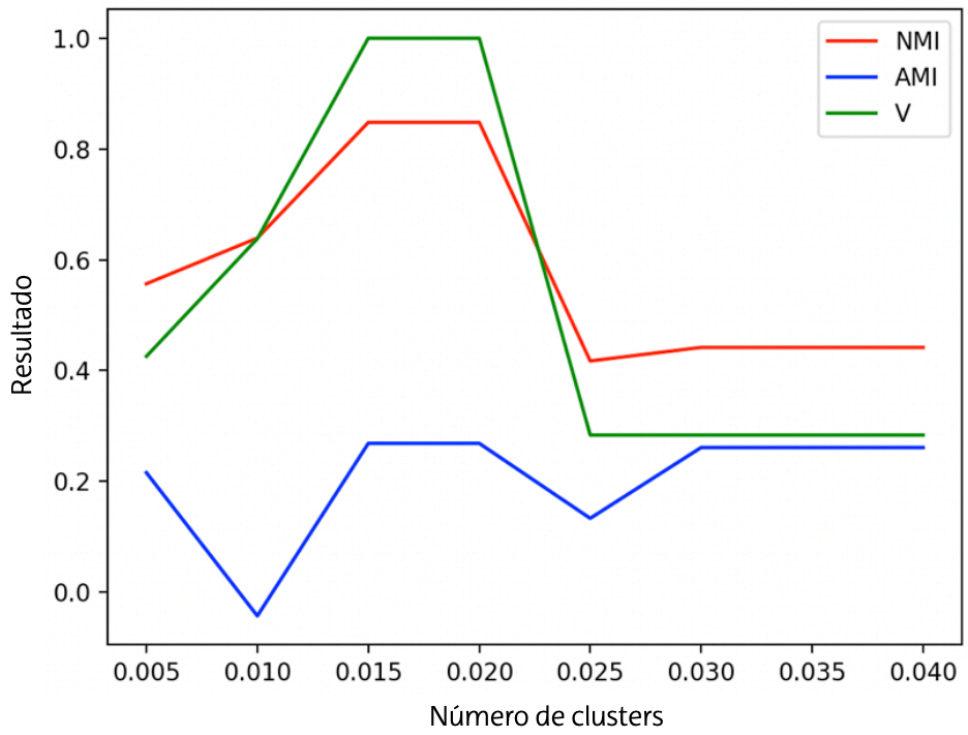


Figura 43: Resultado de dbscan.

En la Figura 43 se observa la relación entre epsilon y el valor obtenido en las métricas. NMI y V siguen obteniendo buenos resultados durante todo el espectro. En cambio, en AMI, se observan los mejores resultados para un valor de epsilon aproximado de 0.015-0.020. Para este resultado se observa una cantidad de ruido considerable, la cual deja a muchos usuarios sin categorizar. Este método no se comporta de forma adecuada.

Una vez desarrollados e implantados todos los métodos y analizados los resultados obtenidos para ellos, el que mejor encaja en la solución es el método aglomerativo. Una buen resultado para este es un valor de 10 clusters. Este es el algoritmo que se implanta en la solución, dejando los otros dos como alternativos. Es posible que cuando aumente la base de datos y las características de los usuarios se definan más algún otro método se comporta de forma más adecuada.

7.3. API

Existen varios puntos relevantes a analizar de cara a comprender su funcionamiento. Es necesario conocer el código y el entorno en el que este se encuentra para poder comprender el funcionamiento de la misma. Además, también se resume brevemente la subida a la nube del código para poder dar respuesta a las solicitudes de la aplicación web.

7.3.1. Definición de un entorno

En primer lugar, se dispone de un entorno virtual en el cual se encuentran todos los paquetes necesarios para la ejecución de la aplicación. Dentro de los paquetes instalados el más importante es Flask-RESTful, el cual es una extensión del micro-framework Flask para construir API REST.

7.3.2. Desarrollo del código

En segundo lugar se crea la base app.py, disponible en la Figura 44. Una vez importadas las librerías mencionadas anteriormente, se define la API y todos los recursos que se utilizan para atender las peticiones de la aplicación web.

```
   /* 1. Preparación de la información                    */
1  importar flask
2  importar flask_restful
3  importar Todo.py recursos
   /* 2. Definición de la aplicación                      */
4  App = Flask
5  Api = Api(App)
   /* 3. Definición de los recursos                       */
6  Api.añadirRecurso(RecursoUsuario)
7  Api.añadirRecurso(RecursoRecomencación).
   /* 4. Definición de la main                            */
8  main()
```

Figura 44: Pseudocódigo de la definición de la API.

Una vez construida la API, se desarrollan los recursos. Esto se muestra en la Figura 45. Se importan las librerías del punto anterior junto a las necesarias para conectarse y trabajar con MongoDB: pymongo y bson o json binario. El segundo y último punto del código es desarrollar los diferentes recursos. En este caso uno para la pestaña de usuario y otro para la pestaña de recomendaciones.

```
   /* 1. Preparación de la información                                    */
1  importar flask
2  importar flask_restful
3  importar pymongo
4  importar bson
   /* 2. Definición del recurso                                           */
5  def recurso Usuario:
6      Base de datos = conectar(url)
7      Respuesta = llamada(Base de datos)
8      Datos = lse preparan los datos (Respuesta)
9      envío datos(Datos)
10 def recurso Recomendaciones:
11     Base de datos = conectar(url)
12     Respuesta = llamada(Base de datos)
13     Datos = se preparan los datos (Respuesta)
14     envío datos(Datos)
```

Figura 45: Pseudocódigo de la definición de los recursos de la API.

7.3.3. Subida a la nube

Una vez construido el código se sube a Heroku. Heroku es una plataforma de servicio de computación en la nube.

Para subir a Heroku es necesario realizar pequeñas variaciones en los archivos de configuración. Una vez realizadas, se sube el proyecto haciendo uso de Github dejándolo disponible en la dirección [38].

En la Figura 46 se muestra un ejemplo de una petición a la API, la cual devuelve el JSON correspondiente. La petición se realiza a la dirección mencionada anteriormente, añadiendo el recurso y el dato requerido "https://‹heroku domain›/todoUsuario/token".

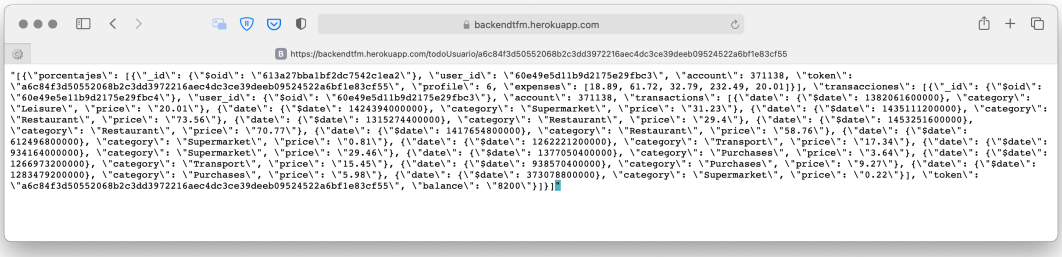


Figura 46: Ejemplo de petición a la API disponible en Heroku.

7.4. Aplicación web

La aplicación NextJs está diseñada para que se adapte a móviles y ordenadores. Para ello, primero se diseña una versión movil y luego se adapta a pantallas con mayor resolución.

Para comprender cómo se estructura la aplicación y cómo se ha desarrollado, resulta interesante conocer la aplicación antes. Por lo que este punto se va a dividir en un apartado relativo a información de funcionamiento para posterior información de desarrollo de la misma.

7.4.1. Pestañas de la aplicación

En la Figura 47 se muestran todas las pestañas disponibles en la aplicación web. En el segundo Anexo se encuentra disponible un manual de la misma con la versión de ordenador.

Figura 47: Aplicación completa versión móvil.

Las pestañas son las siguientes:

- Página inicial. Es la página por defecto. Muestra ciertos datos del banco como descripción, noticias u ofertas generales del mismo.
- Página de usuario. En esta página se muestran los datos del usuario, desde el sueldo disponible en su cuenta hasta un estudio del porcentaje de gasto por categorías o una tabla de transacciones realizadas. A esta página sólo se puede acceder una vez iniciado sesión.
- Página de recomendaciones. En esta página se muestra la información obtenida gracias al Machine Learning. Desde el gasto medio de los usuarios de tu perfil hasta publicidad adaptada al mismo relevante para el usuario. A esta página, al igual que en el caso anterior, sólo se puede acceder una vez iniciado sesión.
- Página sobre nosotros. Página con información de contacto del banco y formulario para envío de correo electrónico.

7.4.2. Estructura del proyecto

La aplicación se organiza como se muestra en la Figura 48. Los puntos más relevantes de analizar son las páginas y los componentes, además de las diferentes configuraciones realizadas.

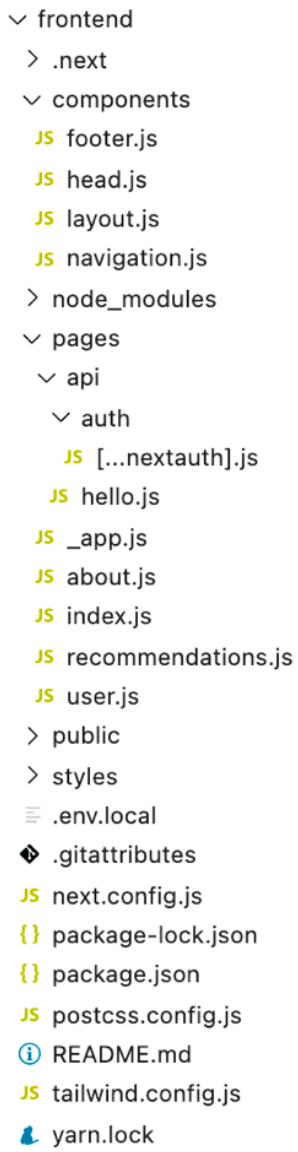


Figura 48: Estructura del proyecto.

En primer lugar se define un layout en la carpeta componentes. El Layout, disponible en la Figura 49, se utiliza para definir la estructura de las páginas, de forma que para cada una de las pestañas se sustituye el valor en el hijo: hijo de la página inicial, hijo de la página de usuario, hijo de la página de recomendados e hijo de la página sobre nosotros.

Todos esos hijos se encuentran definidos en la carpeta de páginas. Esto permite definir en código una vez la cabecera o el pie de página. El código completo se muestra en el Anexo 1.

```
   /* 1. Importar las librerías                                    */
   /* 2. Definición de la función                                  */
 1 Layout = {
 2   Sesion = definir Sesion()
 3   ‹código html›
 4     ‹encabezamiento›
 5         ‹cabecera /›
 6         ‹barra de navegación /›
 7     ‹/encabezamiento›
 8     ‹cuerpo›
 9         ‹hijo /›
10     ‹/cuerpo›
11     pie de página
12   ‹/código html›
13 }
```

Figura 49: Pseudocódigo de la definición del layout.

En cuanto a la estructura que siguen los ficheros hijos se encuentra en la Figura 50. En primer lugar se importan las librerías necesarias. En segundo lugar se define la función con el código html que se va a generar. El código html depende de si existe el usuario o no, es diferente en función del caso. Por último, se define una función asíncrona que se prepara en cada llamada al servidor en el que se solicitan los datos a la API.

```
   /* 1. Importar las librerías                                    */
   /* 2. Definición de la función                                  */
 1 Página = {
 2 preparar Datos()
 3 if existe sesión then
 4 |   Código de no usuario
 5 end
 6 else
 7 |   Código de usuario
 8 end
 9 funcion asincrona
10 }
```

Figura 50: Pseudocódigo de la definición del layout.

La función asíncrona se llama"getServerSideProps ". Esta es la función encargada de llamar a la API y recoger los datos para pasarlos a React a través de props para renderizar la página. Esta función le indica a Next.js que esta página utilizará Server Side Rendering, lo cual indica que se ejecuta la función cada vez que visitemos la ruta en la que se encuentra.

El código completo de algunos de los puntos analizados anteriormente se encuentra disponible en el Anexo. Lo restante, se puede consultar en el enlace [37] disponible en la Bibliografía.

Resulta interesante analizar un par de puntos más de la aplicación. El primero de ellos es la autenticación. Para la autenticación se ha utilizado NextAuth.js. Tal y como se muestra en la 51, permite conectarse con diferentes proveedores como Facebook, Google, etc.

```
import NextAuth from 'next-auth'
import AppleProvider from 'next-auth/providers/apple'
import FacebookProvider from 'next-auth/providers/facebook'
import GoogleProvider from 'next-auth/providers/google'
import EmailProvider from 'next-auth/providers/email'

export default NextAuth({
  providers: [
    // OAuth authentication providers...
    AppleProvider({
      clientId: process.env.APPLE_ID,
      clientSecret: process.env.APPLE_SECRET
    }),
    FacebookProvider({
      clientId: process.env.FACEBOOK_ID,
      clientSecret: process.env.FACEBOOK_SECRET
    }),
    GoogleProvider({
      clientId: process.env.GOOGLE_ID,
      clientSecret: process.env.GOOGLE_SECRET
    }),
    // Passwordless / email sign in
    EmailProvider({
      server: process.env.MAIL_SERVER,
      from: 'NextAuth.js <no-reply@example.com>'
    }),
  ]
})
```

Figura 51: Código de NextAuth.

En este caso se ha conectado con Github, ya que ofrece capacidad de prueba y desarrollo gratuita. Además es la herramienta principal utilizada durante todo el proyecto para crear historial de versiones de código. Pero para el caso de un proyecto en un caso real siemplemente se modificaría el proveedor, en este caso del sector de la banca. Algunos ejemplos de esto serían la autenticación que realizan BBVA o BBK.

En segundo lugar, el estilo de la página se ha definido mediante Tailwind. Tal y como se define en el Análisis de alternativas, Tailwind es un CSS framework orientado al diseño de la aplicación web. En la Figura 52 se muestra un ejemplo de código, el cual es muy similar a CSS. Las características se definen para todos los tamaños de pantalla, por lo que si se quiere hacer adaptable se utiliza, por ejemplo, md: para un ancho mínimo de 768px.

```
<div className="grid grid-cols-1 md:grid-cols-2 gap-4">

    {/* Introducción banco */}
    <div className="max-w-sm md:max-w-5xl md:max-h-5xl bg-white rounded-sm shadow-md overflow-hidden row-span-2">
      <div className="md:flex">
        <div className="md:flex-shrink-0">
        <img className="h-56 w-full object-cover md:h-full md:w-48" src='/cripto.png' alt="Man looking at item at a store"></img>
        </div>
        <div className="p-4 md:p-14 md:h-full">
```

Figura 52: Código Tailwind de muestra.

8. Descripción de tareas

En este apartado se realiza un análisis de la planificación para el desarrollo del proyecto. En él se organizan los puntos a realizar en diferentes fases o etapas. Además se presenta un diagrama de Gantt en el que se visualiza esa planificación realizada.

8.1. Paquetes de trabajo y tareas del proyecto

El proyecto se ha organizado en paquetes de trabajos y tareas, ya que es la metodología típica a aplicar. Los paquetes y sus respectivas tareas son las siguientes:

- **PT1 Definición del proyecto**. En esta primera parte se pone en marcha el trabajo definiendo los objetivos y realizando una búsqueda de información global acerca de todos los ámbitos del proyecto.
    - **T1.1 Definición de los objetivos**. Se define la idea del proyecto y los puntos que ha de cumplir.
    - **T1.2 Búsqueda de información**. Una vez definidos los puntos a realizar se investiga acerca del estado del arte y de las diferentes alternativas para el desarrollo de los mismos.
- **PT2 Diseño de la solución**. Una vez obtenida la información esencial y las alternativas disponibles es necesario realizar el diseño teórico de la solución que cumpla con las necesidades del proyecto. Este diseño se divide en tres tareas:
    - **T2.1 Diseño general**. Se define como se organiza el proyecto a gran escala incluyendo la estructura empresarial y la parte del cliente de los siguientes puntos.
    - **T2.2 Diseño de la estructura empresarial**. Se define como se organiza la empresa respecto al almacenamiento de los datos y cómo se va a aplicar Machine Learning para obtener la propuesta de valor.
    - **T2.3 Diseño de la parte cliente**. Se define como se gestionan esos datos y como se visualizan de cara al cliente.
- **PT3 Implementación de la solución empresarial**. En base al diseño realizado en el punto anterior, se desarrollan los bloques funcionales de la parte empresarial que incluye el lago de datos y el clustering para obtener la propuesta de valor.
    - **T3.1 Procesado de datos**. Se recoge la información de los clientes proveniente de un dataset y se organiza definiendo una serie de características mediante un gobierno de datos.

- **T3.2 Creación del lago de datos**. Se crean las diferentes bases de datos y se conecta con el lago de datos mediante diferentes configuraciones.
- **T3.3 Desarrollo del algoritmo de clustering Kmeans**. Se programa el algoritmo Kmeans para obtener una propuesta de valor de los datos.
- **T3.4 Desarrollo del algoritmo de clustering Aglomerativo**. Se programa el algoritmo Aglomerativo para obtener una propuesta de valor de los datos.
- **T3.5 Desarrollo del algoritmo de clustering DBSCAN**. Se programa el algoritmo DBSCAN para obtener una propuesta de valor de los datos.
- **T3.6 Comparativa de resultados**. Se visualizan los resultados para los tres algoritmos y se escoge el más adecuado para el proyecto.

- **PT4 Implentación de la solución de la parte cliente**. Al igual que en el punto anterior, en base al diseño realizado se implementa la parte que ofrece el servicio al usuario. Este paquete incluyen las tareas:
  - **T4.1 Creación de la API**. Se programan las diferentes funcionalidades requeridas por la aplicación web en una API, la cual es la que se comunica con el lago de datos.
  - **T4.2 Creación de la aplicación web**. Se programa la aplicación adaptada a múltiples dispositivos encargada de visualizar datos de usuarios obtenidos a través de la API.
- **PT5 Evaluación funcional**. Es necesario que todos los bloques anteriores funcionen de forma coordinada, para ello se realizan una serie de pruebas y pequeñas adaptaciones para mejorar los tiempos de ejecución.
  - **T5.1 Revisión de correcto funcionamiento** Una vez unidos los bloques definidos en los pasos anteriores se comprueba que el funcionamiento es el esperado.
  - **T5.2 Mejoras de rendimiento**. Se comprueba el rendimiento de la solución completa y se realizan pequeñas mejoras.
- **PT6 Gestión del proyecto**. Este paquete se realiza durante todo el desarrollo del proyecto, ya que incluye todas las funciones para gestión y control.
  - **T6.1 Seguimiento del proyecto**. Se realizan de forma periódica reuniones de seguimiento y documentación con las directoras de proyecto. En estas reuniones se definen problemáticas encontradas y se buscan soluciones para ellas. Además también se prepara y corrige la documentación a presentar.
  - **T6.2 Preparación de la memoria**. Preparación del documento y de la estructura del mismo.
  - **T6.3 Documentación**. Se dispone de forma pública un repositorio de GitHub con el código. Se redacta el documento final del documento.

Una vez mencionadas los paquetes de trabajo de los que consta el proyecto, el siguiente punto es definir la línea temporal que se sigue para ejecutar cada una de ellos.

8.2. Definición de los paquetes de trabajo y tareas

Para organizar los paquetes de trabajo y sus respectivas tareas en el tiempo, la mejor forma de hacerlo es un diagrama de Gantt. El proyecto abarca el curso académico 2020-2021, pensado para el segundo cuatrimestre. Como días no laborables se establecen los días festivos, vacaciones y los días de exámenes oficiales. En este punto se definen temporalmente cada uno de las fases mediante las siguientes tablas.

8.2.1. Definición del proyecto

EDT	Descripción	Fecha inicial	Fecha final	Horas
PT1	Definición del proyecto	01-02-2021	21-02-2021	**20**
T1.1	Definición de objetivos	08-02-2021	14-02-2021	6
T1.2	Búsqueda de información	08-02-2021	21-02-2021	14

Tabla 13: Tareas de la fase de definición del proyecto.

8.2.2. Diseño de la solución

EDT	Descripción	Fecha inicial	Fecha final	Horas
PT2	Diseño de la solución	22-02-2021	11-04-2021	**100**
T2.1	Diseño general	22-02-2021	28-02-2021	20
T2.2	Diseño estructura empresa	01-03-2021	21-03-2021	50
T2.3	Diseño parte de cliente	22-03-2021	04-04-2021	30

Tabla 14: Tareas de la fase del diseño de la solución.

8.2.3. Implementación de la solución empresarial

EDT	Descripción	Fecha inicial	Fecha final	Horas
PT3	Solución empresarial	05-04-2021	30-05-2021	**220**
T3.1	Procesado de datos	05-04-2021	18-04-2021	60
T3.2	Creación del lago	19-04-2021	25-04-2021	25
T3.3	Desarrollo de Kmeans	26-04-2021	02-05-2021	25
T3.4	Desarrollo de Aglomerativo	03-05-2021	09-05-2021	25
T3.5	Desarrollo de DBSCAN	10-05-2021	16-05-2021	25
T3.6	Comparativa de resultados	17-05-2021	30-05-2021	50

Tabla 15: Tareas de la fase de la implementación de la solución empresarial.

8.2.4. Implementación de la solución de la parte cliente

EDT	Descripción	Fecha inicial	Fecha final	Horas
PT4	Solución parte cliente	31-05-2021	02-08-2021	**120**
T4.1	Creación API	31-05-2021	27-06-2021	50
T4.2	Creación aplicación web	12-07-2021	02-08-2021	70

Tabla 16: Tareas de la fase de implementación de la solución de la parte cliente.

8.2.5. Evaluación funcional

EDT	Descripción	Fecha inicial	Fecha final	Horas
PT5	Evaluación funcional	09-08-2021	05-09-2021	**50**
T5.1	Revisión funcionamiento	09-08-2021	29-08-2021	30
T5.2	Mejora rendimiento	30-08-2021	05-09-2021	20

Tabla 17: Tareas de la fase de evaluación funcional.

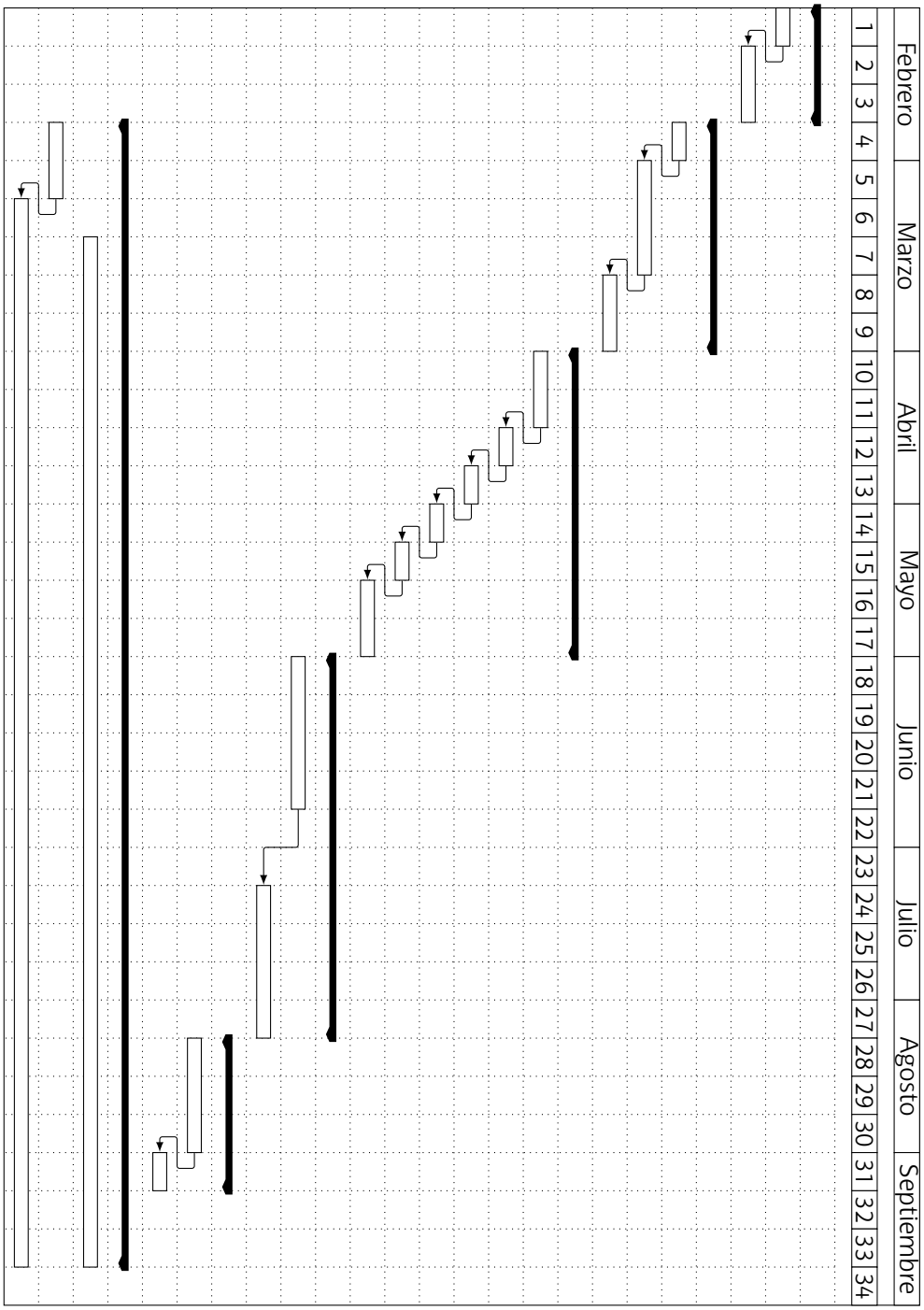
8.2.6. Gestión del proyecto

EDT	Descripción	Fecha inicial	Fecha final	Horas
PT6	Gestión del proyecto	22-02-2021	19-09-2021	**100**
T6.1	Seguimiento	22-03-2021	19-09-2021	35
T6.2	Preparación memoria	22-02-2021	14-03-2021	5
T6.3	Documentación	15-03-2021	19-09-2021	60

Tabla 18: Tareas de la fase de gestión del proyecto.

Una vez definidas las fases y sus respectivos periodos temporales, se visualiza en un Gantt la planificación completa.

8.3. Diagrama de Gantt



9. Descripción del presupuesto

En esta sección se realiza un estudio ecónomico del proyecto mediante un descargo de datos. Como se trata de una investigación, el presupuesto a realizar es simple con una pequeña cantidad de conceptos a tener en cuenta.

El análisis se basa en los siguientes puntos: horas internas, amortizaciones y gastos.

9.1. Horas internas

Para que este proyecto sea viable son necesarios una ingeniera que desarrolle la solución y el documento, y dos directoras encargadas de la organización y toma de decisiones finales.

	Dedicación (h)	Coste horario (€/h)	Coste (€)
Ingeniera Junior	600	25	15000.0
Directora 1	50	50	2500.0
Directora 2	50	50	2500.0
Total			20000.0

Tabla 19: Horas internas.

9.2. Amortizaciones

En primer lugar, se ha trabajado con un ordenador personal de un valor de 1600 €. A este equipo se le estima una 20000 horas de vida útil, es decir, casi 2 años y medio. El software disponible en este equipo para el proyecto es libre, por lo que no supone ningún gasto extra. No serán necesarias licencias en VSCode para la API y la aplicación web, TexStudio para el documento y MongoDB para la base de datos y el Data Lake. Lo mismo sucede con los lenguajes de programación, todos ellos de uso gratuito.

	Coste adquisición €)	Vida útil (h)	Uso (h)	Coste (€)
Macbook Pro	1600.0	20000	600 0	48.0
Total				48.0

Tabla 20: Amortizaciones.

9.3. Gastos

Por último, los gastos no amortizables serán de material de oficina, libros y documentos varios. En este trabajo no se ha requerido de ningún software o equipo específico no reutilizable en otros proyectos.

	Coste (€)
Material oficina	20.0
Libros y documentos	20.0
Total	40.0

Tabla 21: Gastos totales.

9.4. Gastos totales

Para obtener los gastos totales se realiza la suma de las horas internas, las amortizaciones y los gastos. Los gastos totales calculados para el proyecto son de 20088.0 €. Finalmente, se ha añadido una partida de imprevistos del 10 % para prevenir los riesgos comentados en su apartado de Análisis de riesgos, ya que no se puede predecir con exactitud las futuras necesidades.

	Coste (€)
Horas internas	20000.0
Amortizaciones	48.0
Gastos	40.0
Subtotal	20088.0
Imprevistos (10 %)	2008.8
TOTAL	22096.8

Tabla 22: Gastos totales

En la Tabla 22 se muestra el coste final tras incluir la partida de imprevistos. El total es de 22096.8 €, I.V.A incluido.

10. Conclusiones

Tras el desarrollo de este proyecto, se puede afirmar que se han cumplido los objetivos definidos. Se ha desarrollado un lago de datos con información correctamente organizada y analizada que se utiliza para mostrar a usuarios del sector de la banca datos relativos a su cuenta, sus transacciones y una propuesta de valor extra. Esta muestra se realiza mediante una aplicación web adaptada a cualquier dispositivo con navegador. Hasta la fecha, se ha realizado todo el trabajo posible para mejorar el algoritmo que realiza ese análisis para obtener perfiles de los clientes, pero a medida que el número de clientes aumente y sus características sean más concretas los resultados mejorarán.

Por otro lado, es importante destacar que todo este desarrollo se ha realizado teniendo en cuenta el derecho de los propios usuarios a mantener cierta información confidencial. Para ello se han creado diferentes colecciones con ciertos datos ocultos.

Por último, la solución se ha desarrollado de forma que se puede adaptar a otros contextos, no es necesario que se trate del sector de la banca. Esta adaptación se consigue realizando pequeñas modificaciones en los bloques ya construidos. Por ejemplo, variando el dataset por uno con usuarios registrados en un servicio de distribución de contenidos audiovisuales, se puede agrupar a los usuarios en base al tipo de contenido que visualizan y ofrecer recomendaciones basadas en perfiles afines.

10.1. Trabajo a futuro

Tras la realización de este proyecto, debido al interés en este nuevo paradigma estructural, se han definido una series de posibles actualizaciones a realizar.

10.1.1. Ampliación de la arquitectura para múltiples inquilinos

Se trata de completar la arquitectura añadiendo nuevos inquilinos al lago de datos. Esta necesidad se debe a que otros inquilinos pueden encajar mejor en cuanto el almacenamiento de cierta infomación. Este punto se realiza con la llegada de nuevos datos y un análisis de las necesidades en función de los mismos.

10.1.2. Evolución del algoritmo de Machine Learning.

Con la llegada de nuevos datos es posible obtener mejores resultados. Esto se debe a que la eficacia de los algoritmos viene en gran parte definida por los atributos de entrada con los que se entrenan y los patrones en los mismos. El objetivo es realizar un análisis de esos datos y encontrar nuevos patrones y similitudes para los perfiles.

Bibliografía

[1] David Reinsel, John Gantz, John Rydning. *The Digitization of the World From Edge to Core by IDC*. Noviembre de 2018.
https://www.seagate.com/files/www-content/https://www.seagate.com/es/es/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf

[2] Software y Soluciones de Analítica SaS. *Historia del Big Data*.
https://www.sas.com/es_es/insights/big-data/what-is-big-data.html

[3] Openbank. *Diferencia entre Machine Learning, Big Data, Inteligencia Artificial y Data Science*.
https://www.openbank.es/open-news/diferencia-machine-learning-inteligencia-artificial-y-data-science/

[4] Bysidecar. *Que es una Data Driven Company*.
https://bysidecar.com/es/trends/que-es-una-data-driven-company

[5] IBM. *Supervised vs. Unsupervised Learning: What's the Difference?*.
https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning

[6] Paradigma. *Machine Learning, la forma de hacer Big Data inteligente.*.
https://www.paradigmadigital.com/dev/machine-learning-la-forma-big-data-inteligente/

[7] Tesis en red. *Clustering*.
https://www.tesisenred.net/bitstream/handle/10803/3021/gsa3de8.pdf?sequence=3&isAllowed=y

[8] Análisis de datos. *Minería de datos, metodología*.
https://analisisdedatos.net/mineria/tecnicas/clustering/metodologia.php

[9] DelftStack. *Normalizar una columna en Pandas Dataframe*.
https://www.delftstack.com/es/howto/python-pandas/pandas-normalize/

[10] Talend. *¿En qué consiste un data lake?*.
https://www.talend.com/es/resources/what-is-data-lake/

[11] PowerData *Data Lake vs Data Warehouse*.
https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/data-lake-vs-data-warehouse.-veamos-sus-principales-diferencias

[12] Amazon AWS. *What is a data lake?*.
https://aws.amazon.com/es/big-data/datalakes-and-analytics/what-is-a-data-lake/

[13] Aprendeia. *Lenguajes de programación para Machine Learning*.
https://aprendeia.com/lenguajes-de-programacion-para-machine-learning/

[14] Solver. *6 Librerías imprescindibles de Python para Machine Learning*.
https://iasolver.es/6-librerias-de-python-para-machine-learning/

[15] Go4it. *Principales herramientas para el desarrollo de APIs*.
https://www.go4it.solutions/es/blog/principales-herramientas-para-el-desarrollo-de-apis

[16] OpenWebinars. *Django vs Flask*.
https://openwebinars.net/blog/django-vs-flask/

[17] Profile. *Tipos de desarrollo de aplicaciones web*.
https://profile.es/blog/desarrollo-aplicaciones-web/

[18] Gsoft. *¿WEB APPS, APP NATIVA O APP HÍBRIDA?*.
https://www.gsoft.es/articulos/que-necesito-web-apps-app-nativa-o-app-hibrida/

[19] Cantabria TIC. *Introducción a MongoDB*.
http://www.cantabriatic.com/introduccion-a-mongodb/

[20] MongoDB. *MongoDB Atlas Data Lake*.
https://www.mongodb.com/es/atlas/data-lake

[21] MongoDB. *Sample Analytics Dataset*.
https://docs.atlas.mongodb.com/sample-data/sample-analytics/

[22] Medium. *Easy-to-use GDPR guide for Data Scientist. Part 2/2*.
https://www.mongodb.com/es/atlas/data-lake

[23] B12. *¿Qué es Data Masking?*.
https://agenciab12.com/noticia/que-es-data-masking

[24] Medium. *Easy-to-use GDPR guide for Data Scientist.*.
https://korniichuk.medium.com/gdpr-guide-2-7c399b44ba3#b2f9

[25] Ciencia de datos. *¿Clustering y heatmaps: aprendizaje no supervisado*.
https://www.cienciadedatos.net/documentos/37_clustering_y_heatmaps

[26] Ichi. *Paso a paso para comprender el agrupamiento y la implementación de K-means con sklearn*.
https://ichi.pro/es/paso-a-paso-para-comprender-el-agrupamiento-y-la-implementacion-de-k-means-con-sklearn-199389628144086

[27] Aprendemachinelearning. *K-Means en Python paso a paso*.
https://www.aprendemachinelearning.com/k-means-en-python-paso-a-paso/

[28] Kaggle. *Step by Step KMeans Explained in Detail*.
https://www.kaggle.com/shrutimechlearn/step-by-step-kmeans-explained-in-detail

[29] Jarroba. *Selección del número óptimo de Clusters*.
https://jarroba.com/seleccion-del-numero-optimo-clusters/

[30] Towards Data Science. *Hierarchical Agglomerative Clustering Algorithm Example In Python*.
https://towardsdatascience.com/machine-learning-algorithms-part-12-hierarchical-agglomerative-clustering-example-in-python-1e18e0075019

[31] Explonentir. *Ejemplo de uso de DBSCAN en Python para eliminación de outliers*.
http://exponentis.es/ejemplo-de-uso-de-dbscan-en-python-para-deteccion-de-outliers

[32] Esri. *Cómo funciona el clustering basado en densidad*.
https://pro.arcgis.com/es/pro-app/latest/tool-reference/spatial-statistics/how-density-based-clustering-works.htm

[33] Sklearn. *sklearn.metrics.normalized_mutual_info_score*.
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.normalized_mutual_info_score.html

[34] Sklearn. *sklearn.metrics.adjusted_mutual_info_score*.
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.adjusted_mutual_info_score.html#sklearn.metrics.adjusted_mutual_info_score

[35] Sklearn. *sklearn.metrics.homogeneity_completeness_v_measure*.
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.homogeneity_completeness_v_measure.html

[36] Medium. *How I have created and deployed RESTful API using Python and Heroku. Step by step guide*.
https://medium.com/incresco/how-i-have-created-and-deployed-restful-api-using-python-and-heroku-step-by-step-guide-5b9612e6a532

[37] Github. *TFM-next project*.
https://github.com/aliciafl/nextjs-tfm

[38] Heroku. *TFM backend project*.
https://backendtfm.herokuapp.com/

11. Anexo I: Código

En este apartado del documento se incluye el código completo de algunos puntos de interés. La división se basa en los bloques funcionales desarrollados en la solución.

11.1. Clustering

El código para la normalización utilizada en todos los métodos es el siguiente.

Listing 11.1: Normalización

```
def minmax_norm(df):
        return (df - df.min()) / ( df.max() - df.min())
```

11.1.1. Aglomerativo

Listing 11.2: Código aglomerativo

```
variables_dataframe = dataframe.drop(['Unnamed:␣0'], axis=1)
variables_dataframe['Gender']=variables_dataframe['Gender'].astype(int)
variables_dataframe['State']=variables_dataframe['State'].astype(int)
variables_dataframe['Year']=variables_dataframe['Year'].astype(int)
new_df = minmax_norm(variables_dataframe)

# plt.figure(3, figsize=(20,6))
# new_df.plot.scatter(x="State", y="Year")
# plt.show()
X = new_df.iloc[:, [1, 7]].values

plt.figure(1, figsize=(20,6))
dendrogram = sch.dendrogram(sch.linkage(X, method='ward'))
plt.ylabel('Euclidean␣Distance');
plt.show()

model = AgglomerativeClustering( n_clusters=clusters, linkage='ward')
model.fit(X)

labels = model.labels_
no_clusters = len(np.unique(labels) )

new_df['Clusters'] = labels
new_df.to_csv('valoresmedios.csv')
```

11.1.2. Kmeans

Listing 11.3: Código kmeans

```
new_df = dataframe.drop(['Unnamed:␣0'], axis=1)
new_df['Gender'] = new_df['Gender'].astype(int)
new_df['State'] = new_df['State'].astype(int)
new_df['Year'] = new_df['Year'].astype(int)
new_df = minmax_norm(new_df)

X = new_df.iloc[:, [1, 7]].values

# MÉTODO DEL CODO
wcss = []
for i in range(1, 11):
kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
kmeans.fit_predict(X)
# El método inercia devuelve  wcss
wcss.append(kmeans.inertia_)


# PLOT DEL MÉTODO DEL CODO
""" plt.figure(1, figsize=(10,5))
plt.plot(range(1, 11), wcss,marker='o',color='red')
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show() """

# Fitting K-Means to the dataset
kmeans = KMeans(clusters)
labels = kmeans.fit_predict(X)

new_df['Clusters'] = labels

no_clusters = len(np.unique(labels) )
no_noise = np.sum(np.array(labels) == -1, axis=0)

# NUMERO DE ELEMENTOS POR CLUSTER
""" arr = [0]*(no_clusters)
for l in labels:
if l != -1:
arr[int(l)] = arr[int(l)]+1
else:
arr[no_clusters-1] = arr[no_clusters-1]+1

print("array:  "+str(arr)) """

new_df.to_csv('valoresmedios.csv')
```

11.1.3. Dbscan

Listing 11.4: Código kmeans

```
new_df = dataframe.drop(['Unnamed:␣0'], axis=1)
new_df['Gender'] = new_df['Gender'].astype(int)
new_df['State'] = new_df['State'].astype(int)
new_df['Year'] = new_df['Year'].astype(int)
new_df = minmax_norm(new_df)

""" plt.figure(3, figsize=(20,6))
new_df.plot.scatter(x="State", y="Year")
plt.show() """

X = new_df.iloc[:, [1, 7]].values

estimator = PCA (n_components = 2)
X_pca = estimator.fit_transform(new_df)
dist = sklearn.neighbors.DistanceMetric.get_metric('euclidean')
matsim = dist.pairwise(X_pca)
minPts  = 5
A = kneighbors_graph(X_pca, minPts, include_self=False)
Ar = A.toarray()
seq = []
for i,s in enumerate(X_pca):
for j in range(len(X_pca)):
if Ar[i][j] != 0:
seq.append(matsim[i][j])
""" seq.sort()
plt.plot(seq)
plt.show() """

#db = DBSCAN(eps=0.0125).fit(X)  HOMOGENEIDAD PERFECTO 1
#db = DBSCAN(eps=0.025).fit(X)
db = DBSCAN(eps=epsilon).fit(X)

labels = db.labels_

new_df['Clusters'] = labels

no_clusters = len(np.unique(labels) )
no_noise = np.sum(np.array(labels) == -1, axis=0)
```

11.2. API

11.2.1. Aplicación

En primer lugar se encuentra el código en el que se define la aplicación y se marcan los recursos.

Listing 11.5: App.py

```
from flask import Flask
from flask_restful import Api
from resources.todo import Todo

app = Flask(__name__)
api = Api(app)

api.add_resource(Todo, "/todo/<int:id>")

if __name__ == "__main__":
app.run()
```

11.2.2. Recursos

En segundo lugar se encuentra el código con los diferentes recursos que dan respuesta a las peticiones de la aplicación web.

Listing 11.6: Todo.py

```
from flask_restful import Resource
from flask import jsonify
from bson.json_util import dumps
from bson.objectid import ObjectId
import pymongo


#Connection with database
MONGODB_URI = 'mongodb...'
client = pymongo.MongoClient(MONGODB_URI)


class TodoUsuario(Resource):

def get(self, id):
        # Gastos por categoria
        Database = client.get_database('Sandbox')
        SampleTable = Database.Users
        cursor = SampleTable.find( {'token' : id})
        list_cur = list(cursor)

        # Transacciones realizadas
        Database = client.get_database('Bank')
        SampleTable = Database.Accounts
        cursor2 = SampleTable.find( {'token' : id})
        list_cur2 = list(cursor2)
```

```
        prueba = [
        {
                "porcentajes": list_cur,
                "transacciones": list_cur2
        }
        ]
        return dumps(prueba)

class TodoRecomendacion(Resource):

def get(self, id):
        # Datos del Sandbox
        Database = client.get_database('Sandbox')
        SampleTable = Database.Users
        query = SampleTable.find()
        list_query = list(query)

        # Gastos por categoria
        Database = client.get_database('Sandbox')
        SampleTable = Database.Users
        cursor = SampleTable.find( {'token' : id})
        list_cur = list(cursor)

        prueba = [
        {
                "porcentajes": list_cur,
                "perfiles": list_query
        }
        ]

        return dumps(prueba)
```

11.3. Aplicación web

En este apartado se añaden los dos códigos principales. Tal y como se menciona en el Desarrollo de la solución, el resto del código se encuentra disponible en Github.

11.3.1. Layout

Para la estructura del proyecto se utiliza este código:

Listing 11.7: Layout.js

```
import Head from './head'
import Navigation from './navigation'
import Footer from './footer'

import { signIn, signOut, useSession} from 'next-auth/client'

const Layout = ({ children }) => {

const [session] = useSession()

return(
<div className="bg-white␣flex␣flex-col␣min-h-screen␣bg-white␣font-sans">
        <header>
                <Head />
                <Navigation/>
        </header>

        <main className="flex-grow␣mx-10␣md:mx-2␣pt-4␣pb-4">

        <div>{children}</div>

        </main>

        <Footer/>
</div>
)}

export default Layout
```

11.3.2. Autenticación

Para la autenticación se define la siguiente API:

Listing 11.8: auth.js

```
import NextAuth from 'next-auth'
import Providers from 'next-auth/providers'

export default (req, res) =>
NextAuth (req, res, {
        // Configure one or more authentication providers
        providers: [
        Providers.GitHub({
                clientId: process.env.GITHUB_ID,
                clientSecret: process.env.GITHUB_SECRET
        }),
        // ...add more providers here
        ]
        ,
        debug: process.env.NODE_ENV === "development",
        secret: process.env.AUTH_SECRET,
        jwt: {
                secret: process.env.JWT_SECRET,
        },

         // A database is optional, but required to persist
         //accounts in a database
        //database: process.env.DATABASE_URL,
});
```

12. Anexo II: Manual de aplicación

El manual de la aplicación se divide según las pestañas existentes en la aplicación web diseñada con NextJs. La parte que explica el código y como se estructura el proyecto se encuentra disponible en el apartado Desarrollo de la solución, este punto es un simple manual de funcionamiento de la misma.

12.1. Página principal

Se trata de la página inicial que se carga nada más acceder a la página. El estado inicial, disponible en la Figura 53, no tiene usuario registrado. Existen dos forma de iniciar sesión. Por un lado desde la barra de navegación y por otro lado desde el grid, en la entrada de Banco personalizado.

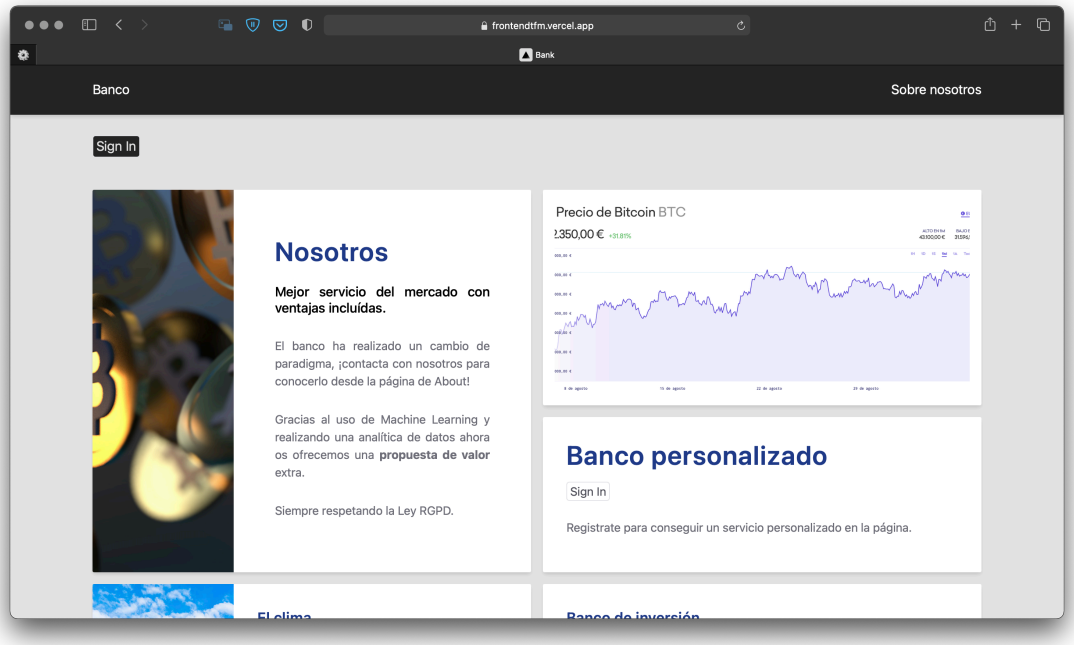


Figura 53: Página principal sin usuario registrado.

Una vez pulsado el botón de Inicio de sesión, se carga la página oficial de Github que se muestra en la Figura 54.

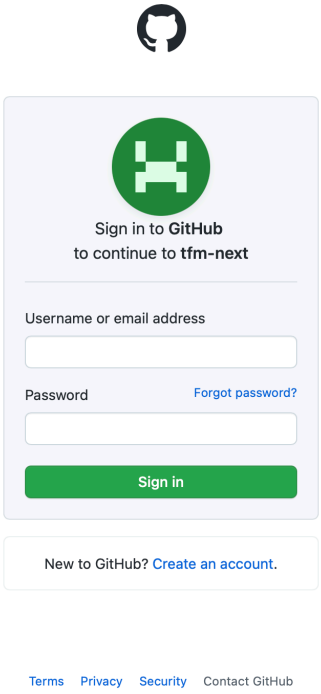


Figura 54: Página de registro a través del proveedor Github.

Tras el inicio de sesión se muestra la pantalla de la Figura 55. La diferencia de la pestaña respecto al estado anterior es la entrada del grid Banco personalizado que se ha sustituido por información de usuario.

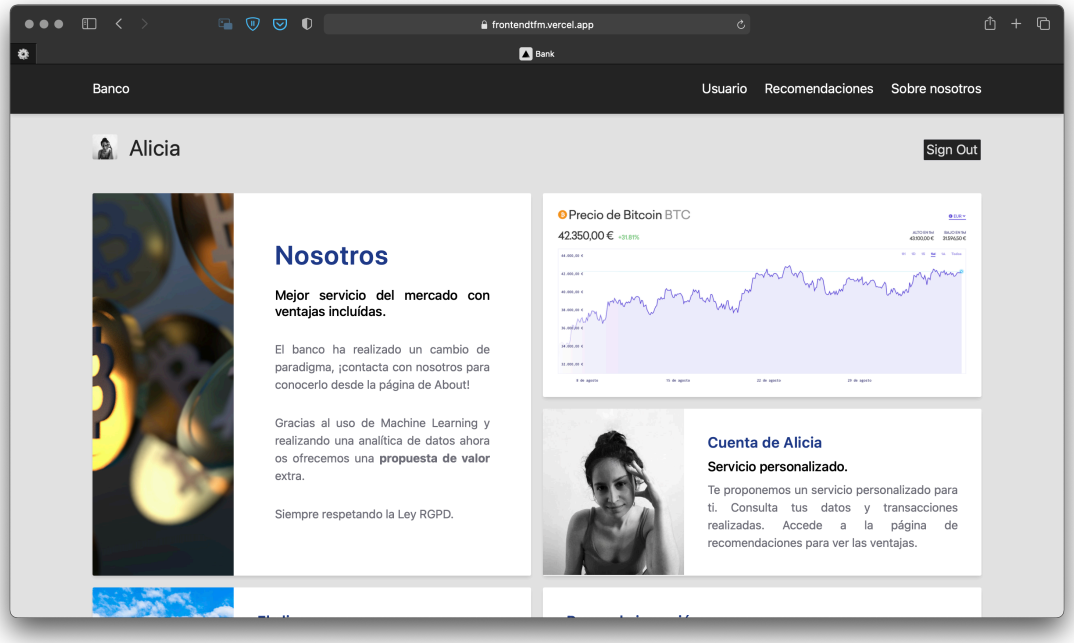


Figura 55: Página principal con usuario registrado.

Es importante destacar que las páginas de usuario y recomendaciones sólo aparecen en la barra de navegación cuando se ha iniciado sesión.

12.2. Página de usuario

Accediendo a la página de usuario, que se muestra en las figuras 56 y 57, se muestran una serie de datos relativos a la actividad y registro del usuario en el banco. Algunos de estos datos son la cantidad de dinero disponible en la cuenta, las transacciones realizadas o un gráfico de porcentajes de gastos por categorías.

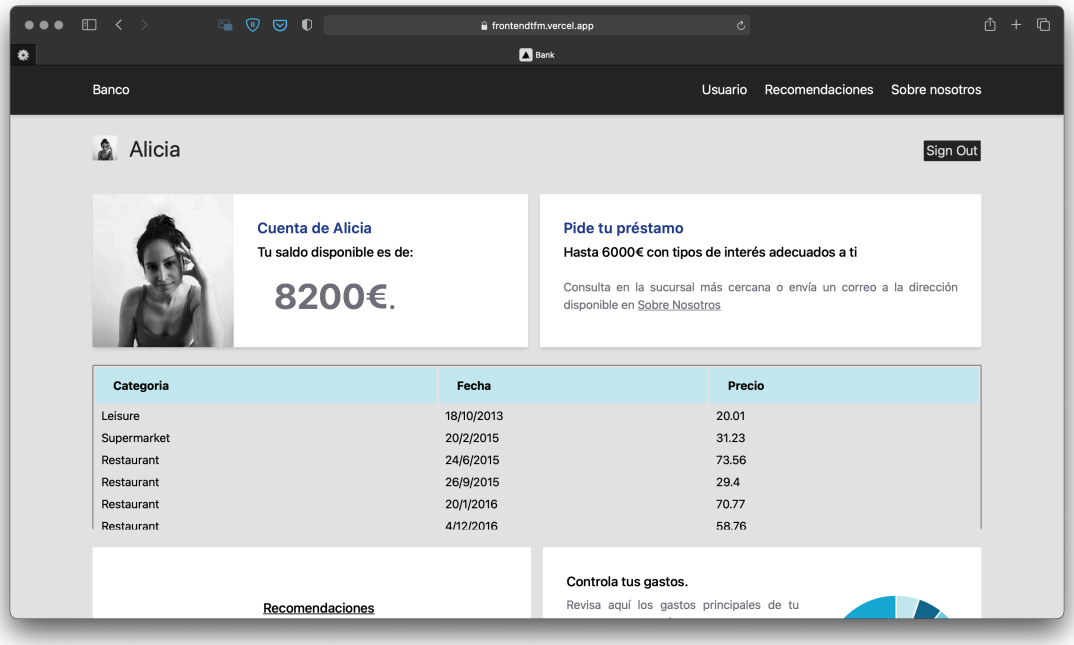


Figura 56: Página de usuario.

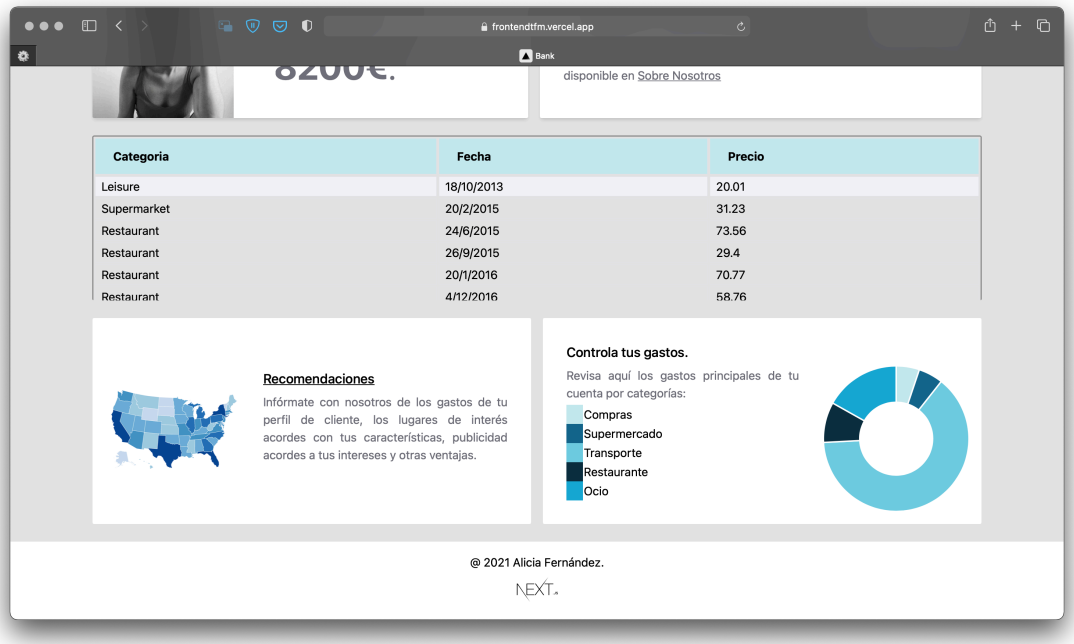


Figura 57: Página de usuario desplazada.

12.3. Página de recomendaciones

La página de recomendaciones, figuras 58 y 59, es aquella que muestra los datos obtenidos a través del perfil del usuario. Por un lado, muestra al usuario su perfil y los gastos por categoría medios del mismo grupo. Por otro lado, se muestran los porcentajes de usuarios en cada perfil. Por último, se muestra publicidad adaptada a la categoría principal de gasto.

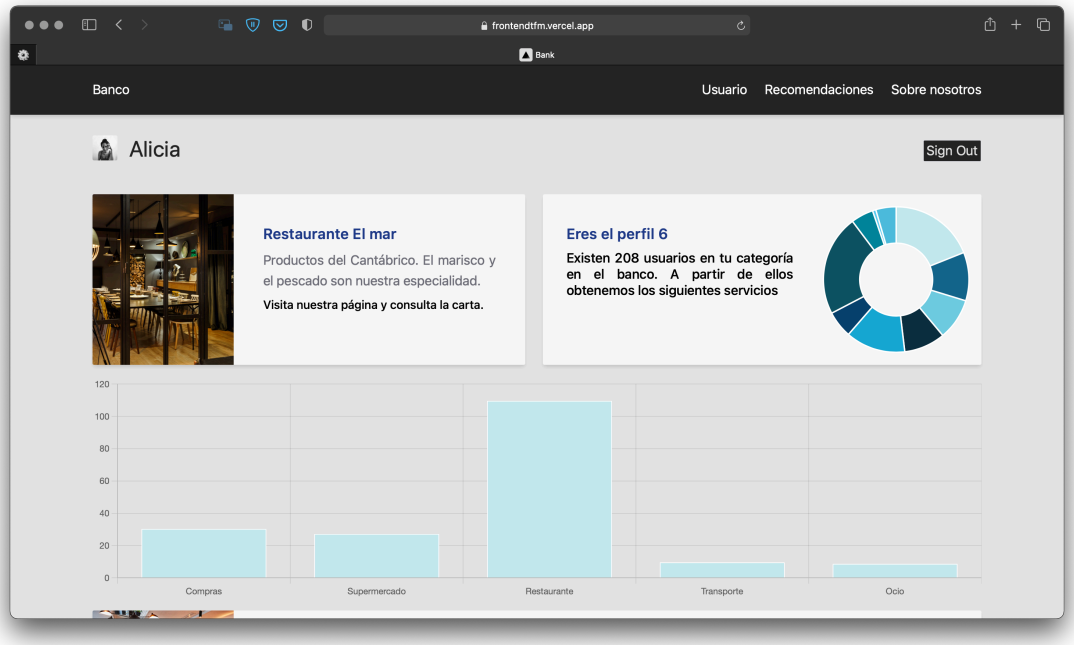


Figura 58: Página de recomendaciones.

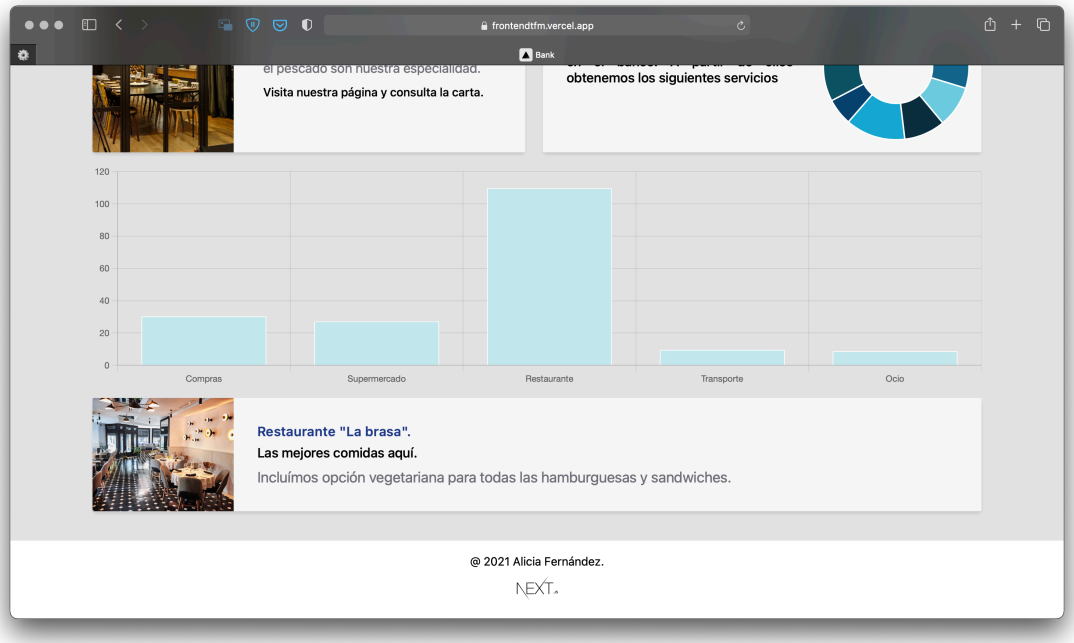


Figura 59: Página de recomendaciones desplazada.

12.4. Página sobre nosotros

La pestaña sobre nosotros, la cual se muestra en la Figura 60, ofrece información relativa al banco, las sedes del banco y teléfono de contacto. Además, se muestra un formulario para enviar un correo con consultas o posibles propuestas. Este te abre unaventana con la aplicación de correo por defecto en el equipo para enviar el mail con información ya detallada.

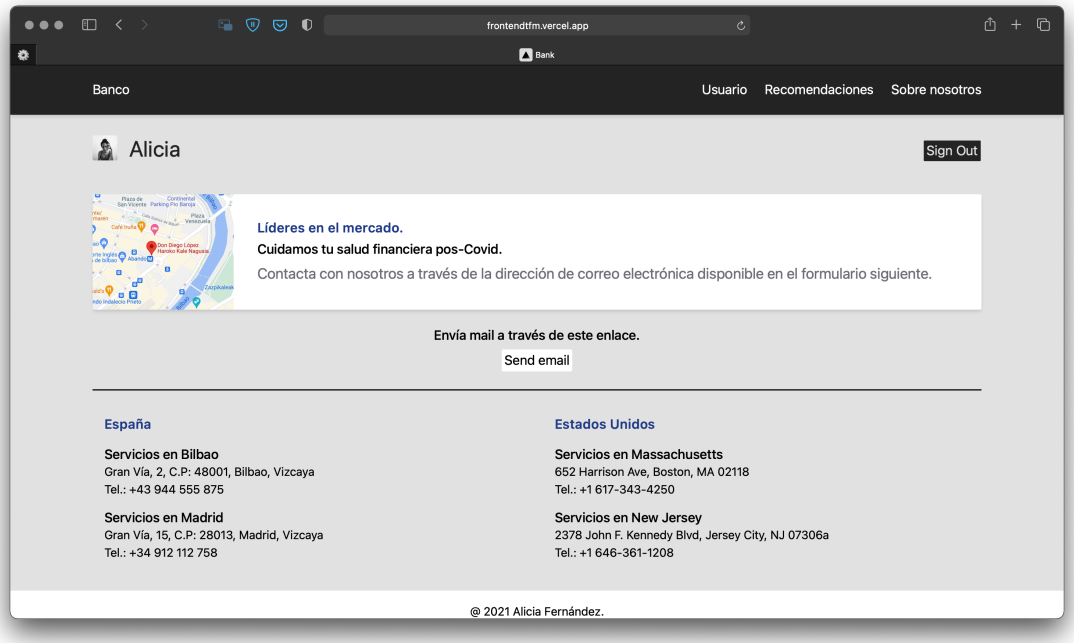


Figura 60: Página sobre nosotros.

El importante resaltar que si se accede a la aplicación web desde un dispositivo móvil, todos los componentes se encuentran adaptados para una correcta experiencia de usuario. En la Figura 61 se visualiza la barra de navegación adaptada.

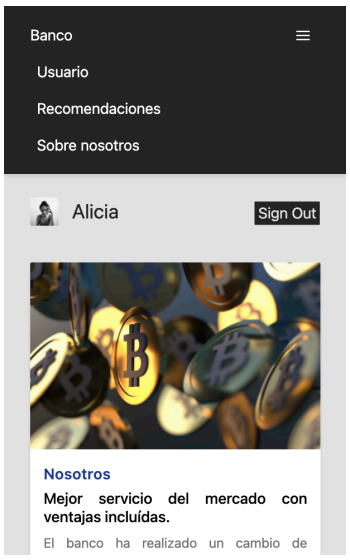


Figura 61: Página adaptada a versión móvil.