VNiVERSiDAD D SALAMANCA

CAMPUS DE EXCELENCIA INTERNACIONAL

Trabajo Fin de Grado de Ingeniería Informática

PLATAFORMA PARA DAR SOPORTE A LA APLICACIÓN DE MACHINE LEARNING EN EL CONTEXTO MÉDICO

Autor: Julia Alonso Sánchez

Tutores: Andrea Vázquez Ingelmo, Alicia García Holgado, Francisco José García Peñalvo.

Julio, 2021

D Francisco José García Peñalvo, D Alicia García Holgado, D Andrea Vázquez Ingelmo, profesores/as del Departamento de informática y automática de la Universidad de Salamanca

CERTIFICAN:

Que el trabajo titulado “Plataforma para dar soporte a la aplicación de Machine Learning en el contexto médico ” ha sido realizado por D. Julia Alonso Sánchez, con DNI 45690856E y constituye la memoria del trabajo realizado para la superación de la asignatura Trabajo de Fin de Grado de la Titulación Ingeniería Informática de esta Universidad.

Y para que así conste a todos los efectos oportunos.

En Salamanca, a 3 de Julio de 2021

VAZQUEZ INGELMO ANDREA - 70918831X
Firmado digitalmente por VAZQUEZ INGELMO ANDREA - 70918831X
Fecha: 2021.07.05 18:01:58 +02'00'

GARCIA HOLGADO ALICIA - 70893684W
Firmado digitalmente por GARCIA HOLGADO ALICIA - 70893684W
Fecha: 2021.07.05 19:55:28 +02'00'

GARCIA PEÑALVO FRANCISCO JOSE - 07862615L
Firmado digitalmente por GARCIA PEÑALVO FRANCISCO JOSE - 07862615L
Fecha: 2021.07.05 20:48:14 +02'00'

Dña. Andrea Vázquez Ingelmo
Dpto. Informática y Automática
Universidad de Salamanca

Dña. Alicia García Holgado
Dpto. Informática y Automática
Universidad de Salamanca

D. Francisco José García Peñalvo
Dpto. Informática y Automática
Universidad de Salamanca

Resumen (español)

El aprendizaje automático es una disciplina que se ha hecho un hueco en muchos ámbitos; entre ellos está el campo de la medicina, donde se utiliza en tareas de diagnóstico, pronóstico, desarrollo de fármacos, etc. Por lo que cada vez más profesionales del campo de la salud se interesan por aprender a preparar datos y entrenar modelos de Machine Learning. No obstante no todos estos profesionales tienen los conocimientos de analítica de datos y tecnología necesarios.

Este es el punto de origen de KoopaML, una aplicación web enmarcada en el contexto del Departamento de Cardiología del Hospital Universitario de Salamanca, cuyo objetivo es permitir que estos usuarios puedan entrenar sus propios modelos, analizar sus datos y realizar tareas sobre ellos sin necesidad de tener conocimientos de programación a través de una interfaz sencilla con un diseño enfocado en el usuario.

Palabras clave: Aprendizaje automático, Diseño centrado en el usuario, Cardiología, Medicina, Interacción persona-ordenador.

Resumen (inglés)

Machine Learning is a discipline which have found their place in many areas. One of these areas is the health sector, where Machine Learning is using in diagnosis, medical prognosis, development of medical products, etc. In this sense, physicians are starting to give attention on how to prepare data and how to train Machine Learning models. However, not every professional have enough knowledge to perform these tasks.
This is the origin of KoopaML, a web application framed in the context of the cardiology department of the University Hospital of Salamanca, whose goal is to allow users to train their own models, analyze their data and perform many other data science tasks without the need of programming skills, by providing a simple interface designed with a user-centered approach.

Keywords: Machine Learning, User-Centered Design, Cardiology, Medicine, Person-computer interaction.

Tabla de contenido

Índice de tablas y figuras

Capítulo 1. Introducción

El aprendizaje automático es una rama de la Inteligencia Artificial centrada en utilizar algoritmos que imiten la forma de aprender de los humanos y permitiendo mejorar el grado de precisión alcanzado (IBM Cloud Education, 2020).

Esta disciplina se ha infiltrado en muchas áreas, una de ellas es la medicina, teniendo utilidad en campos que van desde el diagnóstico y predicción de factores de riesgo, a desarrollo de fármacos (Materials, 2019); aunque la precisión de los resultados está directamente enlazada con la calidad de los datos proporcionados (Pesheva, 2019).

Debido a esto cada vez son más los profesionales médicos que desean conocer más acerca del aprendizaje automático, entrenar su primer modelo de Machine Learning, o ampliar sus conocimientos y habilidades sobre cómo preparar o procesar sus datos para el entrenamiento (Sanyal, 2019). Sin embargo, muchos de ellos no poseen los conocimientos de programación o tecnológicos necesarios para realizar estas tareas.

Este es el origen de KoopaML, una aplicación enmarcada en el contexto del departamento de cardiología del Hospital Universitario de Salamanca, que trata de acercar a los usuarios anteriormente mencionados, con escasos conocimientos de programación y procesado de datos a la disciplina del aprendizaje automático.

Para ello uno de los puntos clave es el desarrollo de una interfaz fácil de comprender, intuitiva y clara, que satisfaga las necesidades y requisitos de los múltiples perfiles que conforman la audiencia objetivo, entre ellos expertos en Machine Learning y usuarios sin apenas conocimientos de ciencia de datos. Por esta razón se ha decidido seguir un enfoque de diseño centrado en el usuario.

Este trabajo muestra todo el proceso de diseño y desarrollo de la aplicación mediante una estructura de capítulos, siendo esta introducción el capítulo 1. El capítulo 2 corresponde a la exposición de los objetivos perseguidos en el trabajo. El capítulo 3 muestra las principales técnicas y herramientas utilizadas para el diseño y desarrollo del sistema. El capítulo 4 se enfoca en una breve descripción de la aplicación desarrollada. El capítulo 5 corresponde a la explicación de los aspectos más relevantes del trabajo, como son la fase de diseño centrado en el usuario y la fase de ingeniería, y otros aspectos como la característica multilingüe de la aplicación, el uso y adaptación de bibliotecas utilizadas y la gestión de los archivos de los usuarios. Finalmente, el capítulo 6 presenta

las conclusiones a las que se ha llegado con la realización de este trabajo. Además, incluye la exposición de líneas futuras de trabajo y de otros trabajos que guardan relación con este.

Además, la presente memoria se complementa con un conjunto de anexos que aportan toda la documentación para evitar que la longitud de esta memoria se extienda más de lo necesario:

- Anexo 1: expone el proceso detallado de diseño centrado en el usuario, acompañado de todos los recursos utilizados y requeridos para comprender el desarrollo y evolución del proceso.
- Anexo 2: expone el proceso detallado de ingeniería acompañado de los diagramas necesarios.
- Anexo 3: incluye el enlace a la documentación del código fuente.
- Anexo 4: incluye el enlace al manual de usuario en formato de vídeo, así como un resumen de las cuestiones que este manual aborda.

Capítulo 2. Objetivos del trabajo

El objetivo principal que persigue este trabajo es la creación de una aplicación web que permita el procesamiento de datos mediante Machine Learning. La aplicación ofrecerá soporte a usuarios sin conocimientos técnicos o con conocimientos muy básicos sobre Machine Learning, pues su funcionamiento se basa en añadir y unir nodos o bloques que representan actividades. La aplicación permitirá también consultar los resultados intermedios y finales obtenidos al aplicar las diferentes tareas sobre los datos. Por tanto, este objetivo principal podría dividirse en los siguientes subobjetivos:

- Permitir la creación, modificación, unión y eliminación de nodos, así como permitir ejecutar el flujo de nodos y obtener los resultados deseados.
- La gestión de archivos, tanto de los archivos subidos por los usuarios como de los ficheros de resultados intermedios generados y los descargados.

Otro objetivo también derivado del objetivo principal es la gestión de los propios proyectos creados, de forma que al usuario se le permita subir nuevos proyectos, descargar los existentes, editarlos, configurarlos y eliminarlos.

Un tercer objetivo derivado del principal es la gestión de la información personal y de acceso de los usuarios de la aplicación a través de acciones como el alta (que ha de ser validada por el administrador), la baja y la modificación de las cuentas de usuario. El administrador tendrá la opción de modificar las cuentas del resto de usuarios.

También ha de controlarse qué actividades pueden realizar los usuarios según su rol. Existirán tres roles: usuario, experto y administrador. Los dos últimos tienen acceso a la funcionalidad de modificar heurísticas.

Un cuarto objetivo es la gestión de las heurísticas en base a las cuales se realizarán predicciones y sugerencias que ayudarán a los usuarios a construir los flujos.

Además, el diseño de la aplicación ha de estar centrado en el usuario, realizando todo el proceso necesario para adecuarlo a las necesidades de la audiencia objetivo.

Finalmente, es necesario que la herramienta sea escalable y flexible, permitiendo añadir múltiples y diversas actividades a las ya existentes.

Además de los objetivos mencionados en los párrafos anteriores a través de este trabajo se pretende alcanzar los siguientes objetivos de índole personal:

- Trabajar con nuevas herramientas y tecnologías desconocidas hasta el momento
- Poner en práctica los conocimientos de ingeniería e interacción persona-ordenador adquiridos durante el grado
- Aprender a trabajar con clientes y usuarios reales

Capítulo 3. Técnicas y herramientas

En este capítulo se exponen las principales técnicas y herramientas que han sido utilizadas para la realización de este trabajo, así como las razones por las que han sido seleccionadas.

3.1. Django (Python)

3.1.1. Python

Para el desarrollo de la aplicación se ha escogido el lenguaje de programación Python, este lenguaje tiene una sintaxis sencilla, muy similar al pseudocódigo, es de código abierto, multiplataforma y multiparadigma (Urquiaga, s.f.).

Además de las características ventajosas mencionadas, Python cuenta con numerosas librerías de matemáticas y ciencias de datos muy útiles para la funcionalidad de aprendizaje automático, como NumPy y ScyPi (GRAPH, s.f.; Garcia Peñalvo & Garcia Holgado, 2017; Garcia Peñalvo & Garcia Holgado, 2017).

3.1.2. Django

Se ha utilizado el framework web Django, ya que permite desarrollar de manera rápida y segura sitios web.

Además Django ofrece gran parte de las funcionalidades básicas que un sitio web requiere, evitando así que el programador tenga que implementarlas.

Django permite que las aplicaciones sean escalables gracias a su filosofía de bajo acoplamiento (Docs, MDN Web, 2021), pues cada parte de la arquitectura de una aplicación es independiente del resto, lo que facilita su modificación.

3.2. MongoDB

En primer lugar, se ha utilizado MongoDB (MongoDB Atlas, s.f.) como base de datos NoSQL.

Se ha hecho uso de una base de datos no estructura por varias razones. La primera razón es la gran variabilidad de datos presentes en la aplicación, si se utilizara una base de datos estructurada existirían muchas tablas con una gran cantidad de campos vacíos.

La segunda razón es que la aplicación ofrece un alto grado de flexibilidad permitiendo al usuario añadir nuevas tareas de diversa índole a la base de datos, el resultado de esto es el almacenamiento de diversas estructuras de datos de distinta naturaleza.

En el caso particular de esta aplicación existen varios campos cuya estructura varía dependiendo del objeto, como son:

- Resultados: este campo pertenece a la entidad Nodo, que representa los nodos de la aplicación. En este campo se almacenan diferentes tipos de datos según el tipo de nodo, estos datos pueden ser: nombres de los archivos intermedios generados como resultado de la ejecución de ese nodo o tipo de datos del *dataframe* utilizado.
- Parámetros: este campo pertenece también a la entidad Nodo. En este campo se almacenan diferentes tipos de datos según el tipo de nodo, estos datos representan la configuración de los parámetros del nodo, pudiendo ser una referencia a un archivo subido por el usuario, un valor numérico, un valor seleccionado de una lista, etc.

Es pues preciso disponer de una base de datos no estructurada para enfrentar con éxito la variabilidad de tipos y estructuras de datos, y lograr así ofrecer un alto grado de flexibilidad al usuario.

3.3. HTML, CSS Y JAVASCRIPT

Para el desarrollo front-end del sitio web se han utilizado los lenguajes de programación HTML, CSS y JavaScript.

Para la definición de la estructura de la página se ha hecho uso de HTML, complementado con la definición de estilos que ofrece CSS, separando así el contenido de la presentación, lo que permite obtener códigos más limpios y sencillos.

Para añadir a la página atributos interactivos, como pueden ser la aparición y desaparición de iconos, los cambios en el aspecto del menú o la creación de nodos de actividades se ha utilizado JavaScript. JavaScript es un lenguaje de programación que puede aplicarse a un documento HTML para ofrecerle el dinamismo anteriormente mencionado.

3.3.1. Bootstrap

Para lograr el desarrollo de un diseño web adaptativo (*responsive design*) se ha utilizado la herramienta Bootstrap, que permite realizar este tipo de diseño a través de componentes reusables y estructuras predefinidas como son las filas, las columnas o las tablas, además de ser una herramienta muy flexible pues permite la personalización de todas las estructuras y componentes (Bacinger, s.f.).

3.4. Metodología

Para el desarrollo de la aplicación se ha aplicado la metodología ágil Scrum, basada en la transparencia, inspección y adaptación (Abellán, 2020).

La parte central de esta metodología son los Sprints, que tienen una duración de entre una y cuatro semanas, y durante los cuales se trabaja en el desarrollo para producir un incremento del producto (VIEWNEXT, 2019). En el caso particular de esta aplicación los Sprints han tenido una duración de una semana, a excepción de los Sprints del desarrollo, los cuales han tenido una duración de 2 semanas. Para gestionar estos Sprints se ha utilizado la herramienta Jira (Atlassian, s.f.).

Otra de las características principales del Scrum es la transparencia y comunicación con el equipo, en este caso la gestión colaborativa del proyecto se ha llevado a cabo a través de Google Drive y Slack, con reuniones periódicas cada semana.

Por otro lado para la realización de los diagramas correspondientes al proceso de ingeniería presentes en el anexo correspondiente, se ha utilizado la herramienta Draw.io.

3.4.1. Gitlab

Para la gestión de versiones del código de la aplicación se ha utilizado Gitlab, un servicio de control de versiones y desarrollo de software colaborativo basado en Git (Wikipedia, 2020), de forma que todos los miembros del equipo tuvieran acceso al historial de las distintas versiones del código en todo momento.

Para integrar de manera eficiente este sistema en el desarrollo se realizaban periódicamente las acciones “commit” y “push”, de forma que el resto de miembros del equipo pudieran revisar la versión más actualizada posible.

Otro aspecto que se tuvo en cuenta a la hora de trabajar con Git fue el desarrollo de código utilizando ramas (branches), de forma que cuando se tenía una versión estable

de la aplicación además de realizar las acciones “commit” y “push” como se expresó en el párrafo anterior, se creaba una nueva rama dedicada a la siguiente funcionalidad a desarrollar, de forma que la versión estable no fuera modificada inintencionadamente. Una vez que se comprobaba el correcto funcionamiento de la nueva funcionalidad desarrollada se fusionaba con la rama de la última versión estable y se repetía el proceso para la siguiente funcionalidad.

3.4.2. Diseño centrado en el usuario

Para el proceso de diseño se ha seguido el enfoque de diseño centrado en el usuario, con el objetivo de lograr satisfacer los requisitos de los distintos perfiles que conforman la audiencia objetivo. Para ello se han utilizado las siguientes herramientas:

- Adobe XD: Se ha utilizado la herramienta Adobe XD [7] para la realización de prototipos digitales. Se ha escogido esta herramienta por varias razones; la primera es que Adobe XD permite crear prototipos de una manera sencilla, conectando las pantallas por las que el usuario puede navegar e indicándole a la herramienta cuál es el evento que desencadenaría esa navegación. La segunda es la facilidad que ofrece de crear bibliotecas de componentes y estilos que pueden utilizarse a lo largo de todo el proyecto. [8]. La tercera es la posibilidad de compartir todo el proyecto a través de un único enlace que se abre en el navegador. También permite a los usuarios a los que se les ha compartido el proyecto realizar comentarios sobre el prototipo a través del enlace.
- Zoom: Se ha utilizado la herramienta Zoom (Zoom, 2021) para la realización del grupo focal en la fase de evaluación con usuarios. Debido a las condiciones socio sanitarias derivadas de la crisis del COVID-19 la evaluación se debía llevar a cabo a través de una plataforma digital, por lo que se escogió Zoom, ya que permite grabar la sesión para futuras revisiones y otorgar el control de tu equipo a otro de los usuarios, lo que resulta de gran utilidad para evaluar prototipos y comprobar de una forma directa cómo enfocan los usuarios el problema que se les plantea.

Capítulo 4. Descripción de la aplicación

KoopaML es una aplicación web que permite a los usuarios entrenar algoritmos y crear modelos de aprendizaje automático, visualizar y explorar datos y diversas actividades más. Para realizar estas tareas no es necesario que los usuarios tengan conocimientos de programación, pues su funcionamiento básico se basa en unir nodos.

Figura 1. Logo aplicación KoopaML

La aplicación presenta una página de inicio, mostrada en la Figura 2, desde la que los usuarios pueden registrarse o iniciar sesión. En esta pantalla también existe una sección que explica de forma breve las principales funciones de la aplicación, denominada “Qué es”, y otra sección en la que se muestra la información de contacto en caso de problema, denominada “Contacto”.

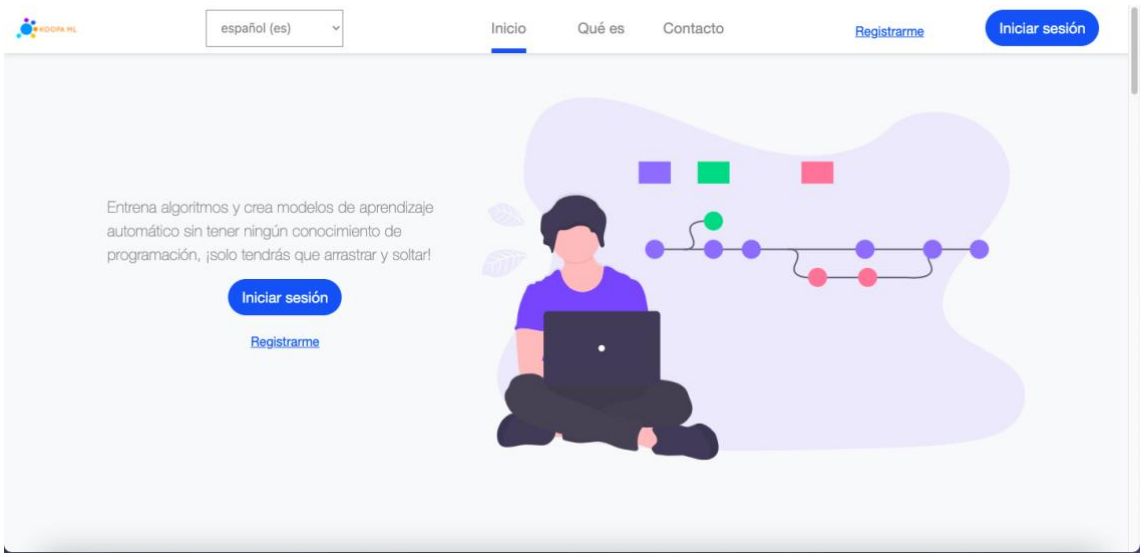


Figura 2. Pantalla inicial KoopaML

La aplicación ofrece a cada usuario, tras iniciar sesión, un área personal, mostrada en la Figura 3, desde la que se le permite crear un nuevo proyecto, subir un proyecto desde su equipo o modificar uno de sus proyectos ya creados. En el caso de los usuarios cuyo

rol sea experto o administrador también podrán acceder a la funcionalidad adicional de modificar las heurísticas que utiliza la plataforma.

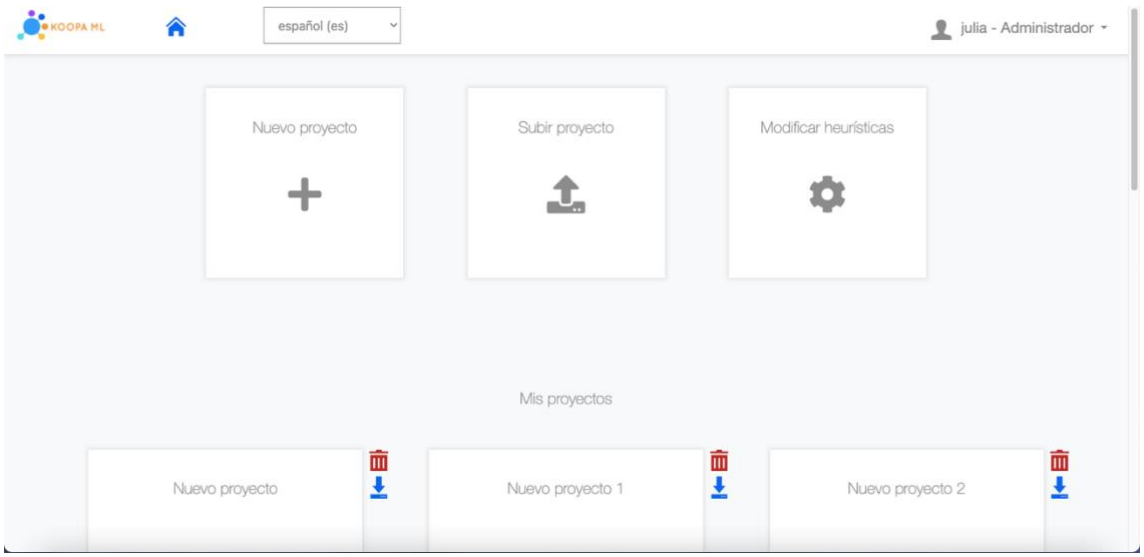


Figura 3. Pantalla área personal Administrador KoopaML

La interfaz de crear un nuevo proyecto o modificar uno existente presenta un lienzo donde se construirá el flujo de nodos que enlace las tareas a realizar y un menú lateral donde se encuentran, organizados por categorías, los nodos que representan las actividades o tareas a realizar, esta disposición puede observarse en las figuras Figura 4 y Figura 5. Las categorías en las que se clasifican los nodos son las siguientes:

- Datos
- Preparación de datos
- Procesado de datos
- Visualización de datos
- Algoritmos de Machine Learning
- Evaluación de resultados

No obstante, la aplicación ha sido creada con el objetivo de ser flexible y escalable, por lo que no se descarta la incorporación de alguna categoría extra, así como la incorporación de la cantidad de nodos o actividades que se deseen.

El menú también ofrece las siguientes opciones:

- Ejecutar
- Ejecutar paso a paso

- Configurar proyecto
- Guardar proyecto

Esta aplicación ofrece gran flexibilidad respecto a las actividades que el usuario puede realizar sobre los datos; una vez se arrastra un nodo desde el menú al lugar de trabajo el usuario puede mover el nodo (cambiar su posición), modificar los parámetros del nodo (en caso de existir) y unirlo a otros nodos con total libertad, permitiendo la creación de flujos de bloques que se adapten a las necesidades específicas de cada usuario.

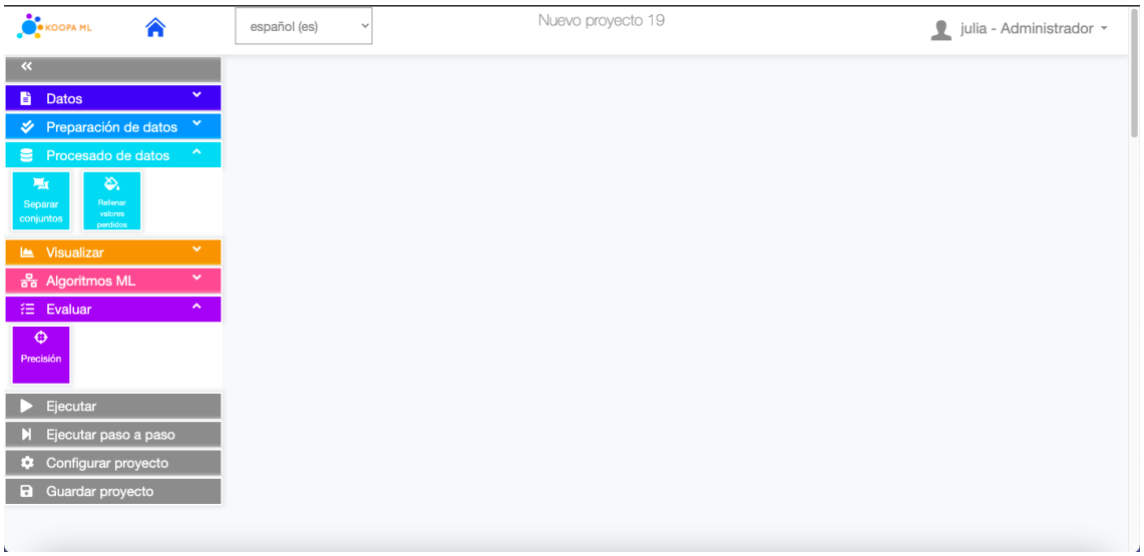


Figura 4. Pantalla de creación nuevo proyecto KoopaML

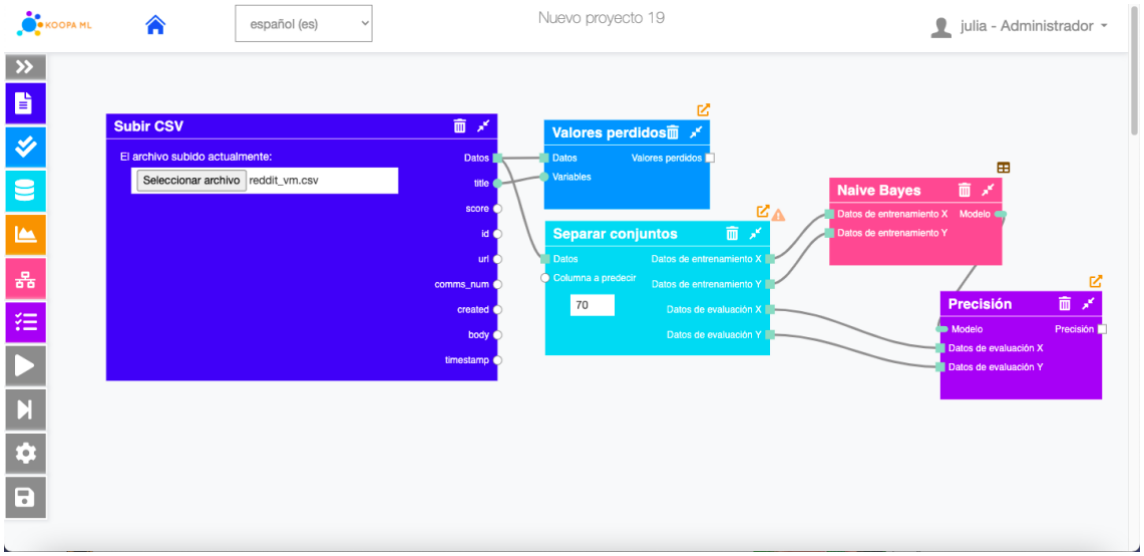


Figura 5. Pantalla de edición proyecto KoopaML

La aplicación también ofrece soporte a los usuarios a través de iconos que muestran errores, advertencias, sugerencias, etc.

Los usuarios pueden descargar los modelos entrenados, así como visualizar los resultados intermedios generados, como se muestra en la Figura 6.

Figura 6. Ventana de visualización de resultados intermedios KoopaML

Los usuarios también tienen acceso a la gestión de su cuenta, como puede observarse en la Figura 7, pudiendo realizar cambios de contraseña o modificaciones en sus datos personales. El administrador además tiene acceso a la información del resto de usuarios, pudiendo realizar acciones como validar o eliminar usuarios como puede observarse en la Figura 8.

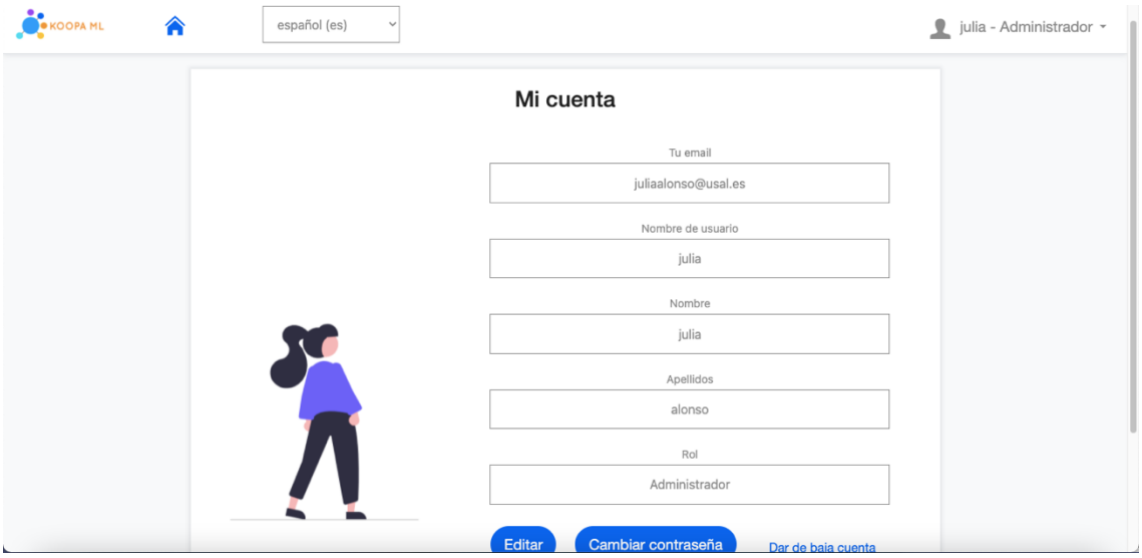


Figura 7. Pantalla “Mi cuenta” KoopaML

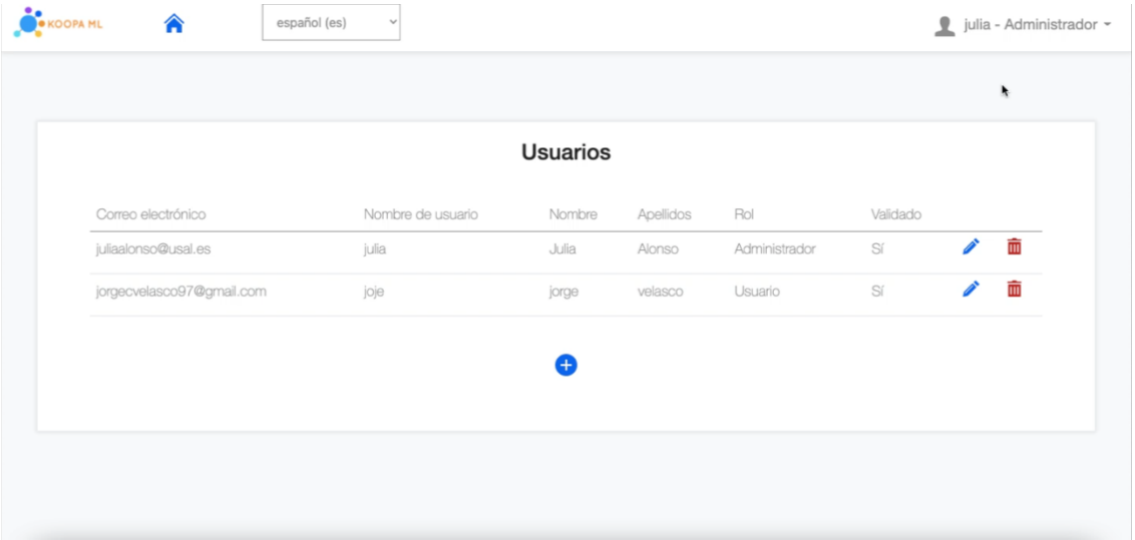


Figura 8. Pantalla "Usuarios" koopaML

Otra funcionalidad de la aplicación es la modificación interactiva de heurísticas que en un futuro servirán para realizar sugerencias dinámicas y automáticas a los usuarios sobre qué algoritmo utilizar en función de los datos de entrada, tarea a realizar, etc.

Los usuarios experto y administrador, que son los únicos roles que pueden acceder a esta funcionalidad, pueden codificar heurísticas que serán representadas a través de un diagrama de flujo de manera automática al pulsar el botón "Actualizar gráfico" como se muestra en la Figura 9. Es posible crear nuevas heurísticas, eliminar y modificar heurísticas existentes, a excepción de la heurística base.

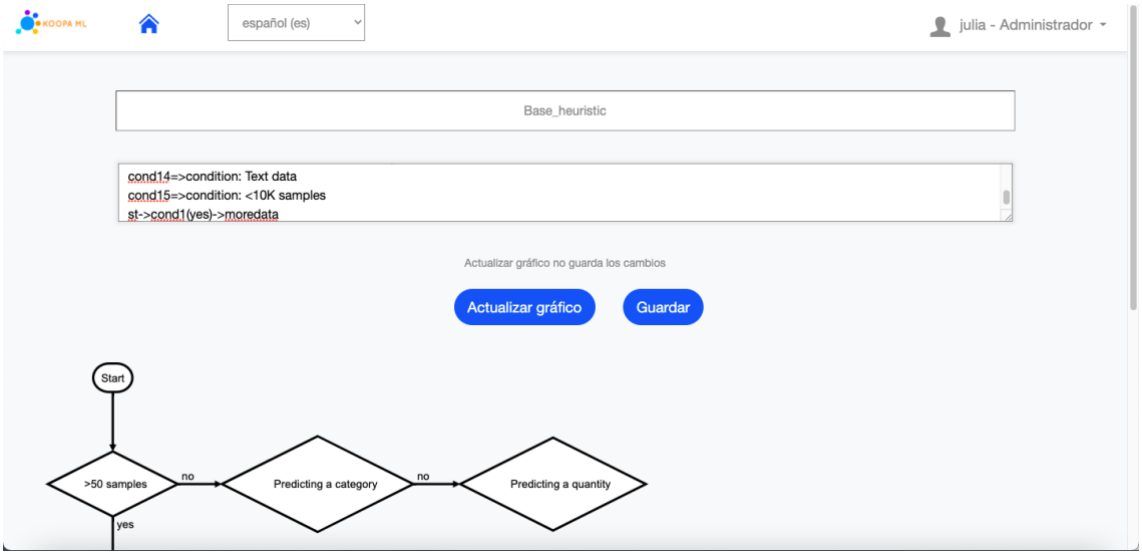


Figura 9. Fragmento de pantalla modificar heurísticas KoopaML

Capítulo 5. Aspectos relevantes

En este capítulo se presentan los aspectos más relevantes en el diseño y desarrollo de la aplicación KoopaML.

5.1. El ciclo de vida

Para el desarrollo de este proyecto se ha seguido la metodología ágil Scrum, explicada de forma breve en el Capítulo 3. Técnicas y herramientas. Esta metodología implica ejecutar Sprints, miniproyectos de corta duración en los que se desarrolla un incremento del proyecto.

Para el desarrollo de esta aplicación se han realizado Sprints cuya duración ha sido de una semana, excepto aquellos destinados al desarrollo del código, cuya duración ha sido de dos semanas.

Al final de cada Sprint, o en la mitad del mismo en los Sprints cuya duración era de dos semanas, se realizaba una reunión con el equipo (los tutores) en el que se analizaba el producto del Sprint a través de una revisión de este, además de realizar una retrospectiva sobre los problemas encontrados, las soluciones escogidas y los resultados.

En el proceso de desarrollo de la aplicación ha incluido el diseño centrado en el usuario, explicado con más detalle en el apartado 5.2. Diseño centrado en el usuario.

Los Sprints realizados:

- Sprint 1: Búsqueda de necesidades y definición de la audiencia.
- Sprint 2: Definición de los escenarios de uso.
- Sprint 3: Realización de los diagramas de casos de uso.
- Sprint 4: Definición de los flujos de proceso y del mapa del sitio.
- Sprint 5: Realización del modelo de arquitectura C4 y del modelo de Entidad-Relación.
- Sprint 6: Aspectos generales del diseño
- Sprint 7: Creación del primer prototipo digital
- Sprint 8: Evaluación del prototipo digital mediante la técnica del grupo focal y realización de las modificaciones necesarias al prototipo.

La parte de diseño centrado en el usuario de estos primeros Sprints corresponde al siguiente ciclo:

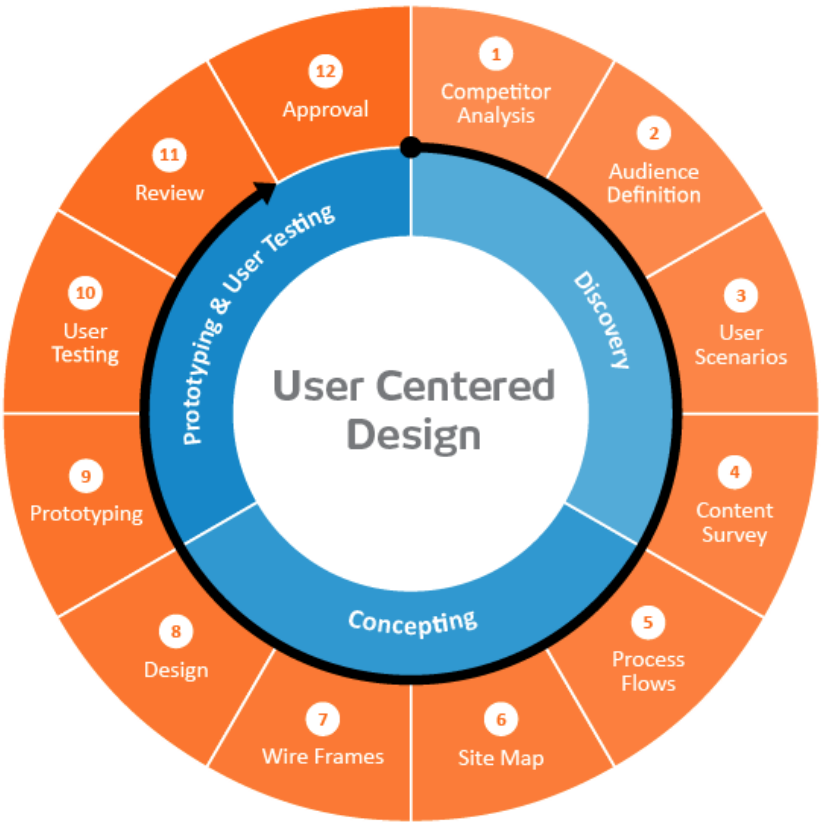


Figura 10. Proceso de diseño centrado en el usuario (Therón, 2020)

- Sprint 9: Creación de las interfaces principales de la aplicación.
- Sprint 10: Implementación de la funcionalidad de gestión de usuarios.
- Sprint 11: Implementación de la funcionalidad de gestión de flujos de nodos.
- Sprint 12: Implementación de la funcionalidad de gestión de heurísticas.
- Sprint 13: Implementación de la funcionalidad de gestión de archivos.

La gestión colaborativa de estos Sprints se llevó a cabo a través de la herramienta Jira.

5.2. Diseño centrado en el usuario

El proceso de desarrollo de la aplicación ha incluido el diseño centrado en el usuario; la documentación que describe todo el proceso de diseño centrado en el usuario y sus resultados pueden consultarse en “Anexo 1. Proceso de diseño centrado en el usuario”. A continuación se exponen de forma breve los aspectos más relevantes de las fases realizadas y sus principales resultados:

1. Fase de exploración

Durante esta fase se ha realizado un trabajo de investigación para poder conocer con más detalle el público objetivo de la aplicación y sus necesidades. Como resultado se ha obtenido la descripción de los usuarios en forma de Personas (Qiany, 2020), así como la representación de los escenarios en los que se hará uso de la aplicación (Interaction Design Foundation, s.f.).

Para contextualizar el marco de la aplicación y la búsqueda de necesidades es importante mencionar que existen muchas herramientas cuyo objetivo es permitir el entrenamiento de modelos de aprendizaje automático.

Se podrían organizar en tres categorías:

- Herramientas para desarrolladores y científicos de datos, las cuales proveen bibliotecas para la creación de aplicaciones Machine Learning.

  Ejemplos de estas herramientas: TensorFlow (Martín Abadi, 2016), Apache Mahout (MAHOUT, 2021), librerías de Python como Py-Torch (PyTorch, s.f.), Scikit-learn (scikit-learn, s.f.) o Keras.io (Keras, s.f.) y servicios en la nube como Google Colab (Google, s.f.).

- Aplicaciones cuyo público objetivo son expertos, pero ofrecen soporte y herramientas para usuarios no expertos. Particularmente hay varias aplicaciones de este tipo que permiten la definición visual de los procesos de aprendizaje automático. Ejemplos de estas herramientas: Weka (Weka 3: Machine Learning Software in Java , s.f.), que tiene una interfaz visual denominada KnowledgeFlow, la cual permite a los usuarios especificar un flujo de datos conectando gráficamente componentes que representan fuentes de datos, herramientas de pre-procesamiento, algoritmos, visualizaciones…

  Otro ejemplo de estas aplicaciones es RapidMiner Studio (rapidminer, 2021), que contiene herramientas visuales de diseño de flujos para crear flujos de aprendizaje automático en los que cada paso está documentado para lograr una transparencia completa.

  Otro ejemplo es KNIME Analytics Platform (KNIME, s.f.). Esta aplicación provee herramientas para la creación visual de flujos para análisis de datos a través de una interfaz gráfica, sin necesidad de programar.

- Aplicaciones cuyo objetivo es ofrecer conocimiento sobre el aprendizaje automático en escuelas e institutos. Estas aplicaciones proveen herramientas para ayudar a usuarios no expertos, como niños, en el desarrollo de un flujo básico de aprendizaje automático usando una interfaz visual.

  Ejemplos de estas aplicaciones: LearningML (LearningML, 2021) y Machine Learning for Kids (Machine Learning for Kids, s.f.). Ambas basadas en un flujo simple que es utilizado para entrenar modelos y que permite una integración con Scratch para usar los modelos entrenados.

Se puede comprobar pues la existencia de multitud de aplicaciones enfocadas a hacer más sencilla la aplicación de algoritmos de aprendizaje automático, muchas de ellas también ofrecen un soporte educativo para comprender mejor flujos complejos.

No obstante, la aplicación KoopaML ha sido concebida como una herramienta personalizada, con requisitos específicos relacionados con el campo de la salud y con gran interés en ofrecer una experiencia educativa a aquellos usuarios que no poseen conocimientos técnicos.

Esta fase de exploración se corresponde con los dos primeros Sprints mencionados en el apartado 5.1. El ciclo de vida.

1.1. Búsqueda de necesidades

Los usuarios utilizaban una herramienta menos flexible e intuitiva para entrenar algoritmos y modelos de aprendizaje automático. Solicitaban una herramienta con una interfaz intuitiva que permitiera realizar esas tareas de un modo más personalizado, además de otros requisitos recogidos en los escenarios de uso documentados en el apartado 5.2. Fase ingeniería, por lo que, a pesar de la existencia de múltiples programas que acercan la aplicación del aprendizaje automático a los usuarios, se decidió realizar una nueva aplicación que pudiera satisfacer todos los requisitos solicitados.

Esta fase se desarrolló durante el primer Sprint (semana 1).

1.2. Definición de la audiencia

El objetivo de la definición de la audiencia es lograr desarrollar soluciones adecuadas a las necesidades, objetivos y situaciones de los usuarios que las van a utilizar. Para ello es de gran importancia comprender a los usuarios que conforman la audiencia

objetivo, y por esta razón se ha escogido utilizar el método Persona de Alan Cooper (Qiany, 2020) para la definición de este público objetivo. Para aplicarlo de manera correcta se ha seguido una guía de 10 pasos (Rikke Friis & Yu Siang, 2021) basada en los pasos que expone Lene Nielsen en su artículo Personas (Nielsen L. , 2010).

Esta fase se desarrolló, al igual que la anterior, durante el primer Sprint (semana 1).

Como resultado se han creado las Persona mostradas en la Tabla 1.

Cabe destacar que esta tabla es el resultado de la iteración final correspondiente a esta fase, no de la inicial, si se desea consultar la definición inicial de Personas o el proceso de iteraciones a través del cual evolucionó esta definición debe consultarse el anexo “Anexo 1. Proceso de diseño centrado en el usuario”.

Nombre	**Cuáles son las necesidades de la persona**	**Cuál es la situación**
Genoveva	Entrenar algoritmo de aprendizaje automático sin tener apenas conocimientos de programación ni ML	Médica veterana, en su hospital quieren incorporar herramienta de aprendizaje automático. Ella siente interés y considera que es una ventaja pero no sabe usarla.
Javier	Visualizar y analizar los datos de los que dispone en su trabajo.	Médico que a menudo trata con una gran cantidad de información que desea poder organizar, analizar y visualizar.
Aarón	Aprender sobre el análisis de datos y su utilidad y ventajas en la medicina	Médico que quiere aprender nuevas técnicas y herramientas que le ayuden en su profesión.
Sheila	Poder usar una aplicación de aprendizaje automático que	Médica que da gran importancia de la

	muestre la visualización en detalle y con ayudas de los resultados	visualización e interpretación de los datos.
Leticia	Aprender sobre los algoritmos de aprendizaje automático de manera práctica	Médica interna residente que ha visto a su tutor usar ML de reconocimiento de imágenes en el trabajo y se cuestiona si podría sería mejor utilizar otro algoritmo y cuándo es más adecuado usar uno u otro
Laura	Aprender los fundamentos y funcionamiento de ML para comprender la teoría que está estudiando de una manera rápida	Estudiante de quinto curso de medicina que está estudiando sobre ML aplicado a la medicina
Juan	Poder personalizar los parámetros y heurísticas de una aplicación de ML y aprender de los resultados de sus pruebas	Experto en ML que quiere poder personalizar los parámetros y heurísticas de una aplicación de aprendizaje automático

Tabla 1. Definición definitiva de Personas

1.3. Escenarios de uso

Para la definición de escenarios de uso principalmente es necesario identificar: quién es el usuario, por qué los usuarios buscan el producto, qué objetivo tiene el usuario, cómo se usará la aplicación y otros detalles con relación a los usuarios y sus experiencias con el producto (Interaction Design Foundation, s.f.). Muchas de estas cuestiones han sido resueltas gracias a la aplicación de la técnica Personas (Qiany, 2020) en el apartado anterior. La representación de estas cuestiones en forma de escenarios de uso puede observarse en: “Anexo 1. Proceso de diseño centrado en el usuario”.

Esta fase se llevó a cabo durante el segundo Sprint (semana 2).

2. Fase de conceptualización

En esta fase se ha profundizado en la organización de la aplicación web, definiendo los flujos de proceso de las tareas principales y realizando de manera conceptual la distribución y mapa del sitio. El resultado de esta fase ha sido la creación de un prototipo digital que refleja todos los aspectos de la investigación anterior, acompañado de un estudio sobre el diseño de la aplicación.

Esta fase de conceptualización se corresponde con los Sprints cuarto y sexto.

2.1. Flujos de proceso

Para el análisis de tareas y posterior representación en forma de flujo de proceso se ha utilizado el método "divide y vencerás" o descomposición de tareas complejas en tareas más simples, en concreto se ha escogido representarlo a través de un análisis jerárquico (HTA) (Therón, 2020).

La representación de todos los flujos de proceso puede observarse en: "Anexo 1. Proceso de diseño centrado en el usuario".

En la Figura 11 se muestra como ejemplo el flujo de proceso de la tarea crear proyecto, que por su dificultad se ha dividido en varias subtareas: iniciar sesión, seleccionar crear nuevo proyecto, agregar los elementos, configurar los parámetros de los nodos, ejecutarlo y configurarlo. La subtarea número 3, agregar elementos, se divide a su vez en dos opciones de tareas, agregar nodos o agregar uniones. A su vez la subtarea 5, ejecutar, también se divide en dos opciones de tareas, ejecutar el proyecto paso a paso o ejecutar todo el proyecto.

El resultado de esta fase corresponde al cuarto Sprint (semana 4).

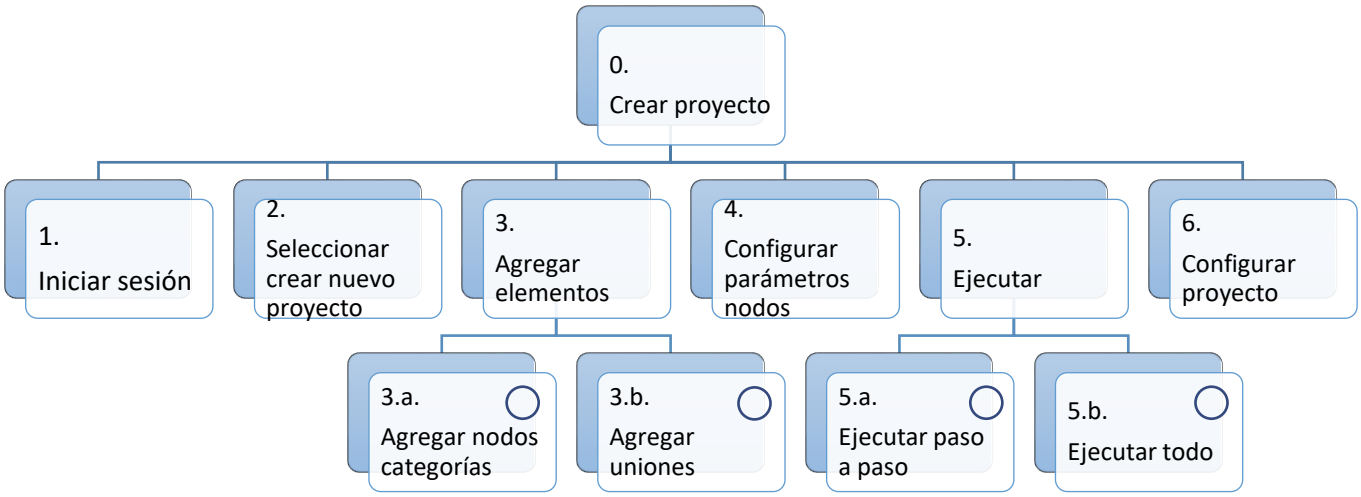


Figura 11. Flujo de proceso crear proyecto

2.2. Mapa del sitio

El mapa del sitio muestra la estructura en forma de secciones de la aplicación, como puede observarse en la Figura 12, partiendo del inicio los usuarios pueden registrarse, obtener más información sobre la aplicación e iniciar sesión. El inicio de sesión permite a los usuarios acceder a su área personal, comenzar un nuevo proyecto, subir un proyecto o modificar heurísticas.

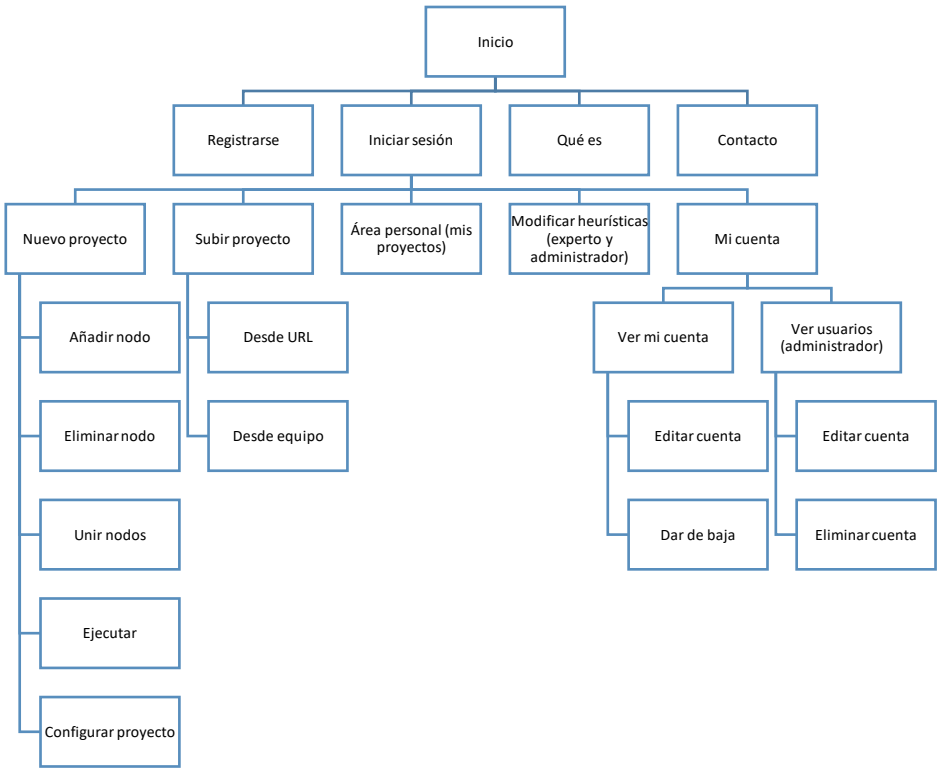


Figura 12. Mapa del sitio

El resultado de esta fase, al igual que en el caso anterior, corresponde al cuarto Sprint (semana 4).

2.3. Diseño

Todos los aspectos a considerar sobre el diseño se encuentran documentados en "Anexo 1. Proceso de diseño centrado en el usuario", a continuación se exponen con brevedad los puntos más importantes:

- Las distintas categorías que engloban las tareas están representadas cada una por un color, de forma que el usuario pueda identificar a qué categoría pertenece un nodo de un solo vistazo al flujo.
- El color principal utilizado es el azul, pues representa confianza, seguridad, productividad y está fuertemente relacionado con las dos disciplinas en las que se encuadra este proyecto: tecnología y medina. En torno a este color ha sido generada la paleta de colores siguiendo la guía sobre Material Design de Google (Google, s.f.).
- Las tipografías utilizadas son sans serif. La tipografía escogida para el cuerpo de la aplicación es Helvética Neue, pues se pretende fomentar la sencillez y claridad de la interfaz de la aplicación.
- Las salidas y entradas de los nodos difieren en su forma dependiendo del tipo de socket al que pueden ser conectados, las conexiones que representan datos tienen forma cuadrada, aquellas que representan variables circular, y las que representan modelos ovalada. Esto es así para facilitar a los usuarios el conocimiento de las posibles conexiones entre nodos, de forma que visualmente puedan tener conocimiento sobre qué salidas pueden conectarse a qué entradas.

  Los sockets también cambian su color a verde cuando se unen a otro socket.
- Cada nodo está representado por un círculo del color de la categoría a la que pertenece y por un icono que representa su funcionalidad. El objetivo de esto es que la interfaz sea lo más sencilla e intuitiva posible, además de tratar de reducir la carga cognitiva del usuario (Cantú, 2018) priorizando el reconocimiento de un nodo sobre el esfuerzo de recordar ese nodo (sexto

principio de usabilidad de Jacob Nielsen: “reconocer mejor que recordar” (Nielsen J. , 2020)).

3. Fase de prototipo y evaluación con usuarios

En esta fase se ha desarrollado también de forma cíclica, a través de varios Sprint, una evaluación con usuarios sobre un prototipo de la aplicación, posteriormente se utilizan los resultados de la evaluación para modificar los puntos necesarios, con el objetivo de solucionar problemas encontrados o mejorar ciertos aspectos.

Los resultados de esta fase se corresponden con el séptimo y el octavo Sprint. Durante el séptimo Sprint se realizó el primer prototipo digital a través de la herramienta Adobe XD (semana 7), durante el octavo Sprint se realizó la evaluación de este primer prototipo con usuarios a través de la técnica del grupo focal, así como se realizaron las modificaciones necesarias, dando como resultado un segundo prototipo digital que incorporaba soluciones a los problemas detectados en la evaluación (semana 8).

3.1. Prototipo digital y evaluación con usuarios

Las imágenes que muestran el primer prototipo digital pueden consultarse en el Anexo 1, el segundo prototipo digital, resultado de los cambios producidos tras la evaluación con usuarios puede ser consultado a través del siguiente enlace: https://xd.adobe.com/view/a506e311-743f-4af3-9659-e5a259aabf1a-f055/.

Este apartado se centrará en las interfaces sobre las que más problemas se encontraron, ilustrando la evolución de estas interfaces a través de imágenes que muestran el antes y el después de la evaluación con los usuarios.

La evaluación del primer prototipo por los usuarios fue llevada a cabo siguiendo la técnica de grupo focal a través de la plataforma Zoom. La muestra de usuarios representantes de la audiencia objetivo estuvo conformada por siete personas, tres cardiólogos con interés en el aprendizaje automático (médicos), tres profesionales del ámbito de la ingeniería y la física con altos conocimientos en el tratamiento y análisis de datos (experto en ciencia de datos), y un alumno de informática que se encontraba realizando un trabajo fin de grado relacionado con el aprendizaje automático.

Debido a las condiciones socio sanitarias derivadas de la crisis del COVID-19 la evaluación se llevo a cabo a través de la plataforma de Zoom y tuvo una duración de una hora. Se solicitó el consentimiento de grabación del proceso de evaluación a los usuarios con el objetivo de recopilar la máxima cantidad de información posible a través de posteriores revisiones y analizar las respuestas y el comportamiento de los usuarios de una manera más exhaustiva. Los usuarios dieron su consentimiento expreso para la grabación.

Durante la evaluación se les compartió la pantalla a los usuarios para que pudieran seguir toda la navegación en tiempo real, realizándoles diversas preguntas sobre cada una de las pantallas y sobre la navegación entre ellas. Estas preguntas no presentaban juicios de valor ni afirmaciones con carácter ideológico, ni orientaban o condicionaban la respuesta del usuario (Vila, 2012). Los usuarios participaron activamente a través de respuestas y sugerencias, así como interactivamente, pues en ocasiones se les ofreció el control de la navegación para poder observar su interacción con la aplicación.

El proceso de evaluación, incluyendo los detalles del guion utilizado y los resultados obtenidos, pueden consultarse en Anexo 1, asimismo, el proceso y resultados se han presentado en las VII Jornadas Iberoamericanas de Interacción Humano-Computador que tendrán lugar en Sao Paulo (Brasil). A continuación se exponen los problemas, aspectos positivos y comentarios más relevantes que surgieron de esta evaluación, así como las imágenes del primer y segundo prototipo que ilustran la evolución de las interfaces:

- Pantalla nuevo proyecto:
    - Aspectos negativos:
        - “No queda claro cómo proceder para crear y unir nodos la primera vez que se utilice la herramienta si no se ha utilizado una herramienta similar antes. Invita a seguir una estrategia ensayo-error. La primera vez que uno entra en esta pantalla piensa “aquí la primera fallo seguro”. No es auto explicativo por completo”. (Luis Miguel, médico)
    - Sugerencias

        - “La primera vez que accedes estaría bien una pequeña directriz o instrucción hasta que arrastras o haces algo tú que se quita” (Acho, experto en ciencia de datos)
        - "La barra de herramientas mostrar expandida al inicio en lugar de minimizada” (Eduardo, médico)
    - Aspectos positivos
        - Fácil la forma de llegar a la pantalla (los usuarios pudieron llegar a esta pantalla sin ningún problema)
        - “Pantalla intuitiva” (David, médico; Acho, experto ene ciencia de datos; Eduardo, médico; Luis Miguel, médico)
        - “Apariencia buena” (Luis Miguel, médico; Acho, experto en ciencia de datos)
        - “Flexible, puedes añadir varios nodos y en el orden que quieras” (Antonio, Acho, Jesús, expertos en ciencia de datos)

A continuación se muestran las imágenes de los dos prototipos digitales realizados, con el objetivo de ver su evolución tras la evaluación con el grupo focal de usuarios.

- Primer prototipo (antes de evaluación) ilustrado en la Figura 13.

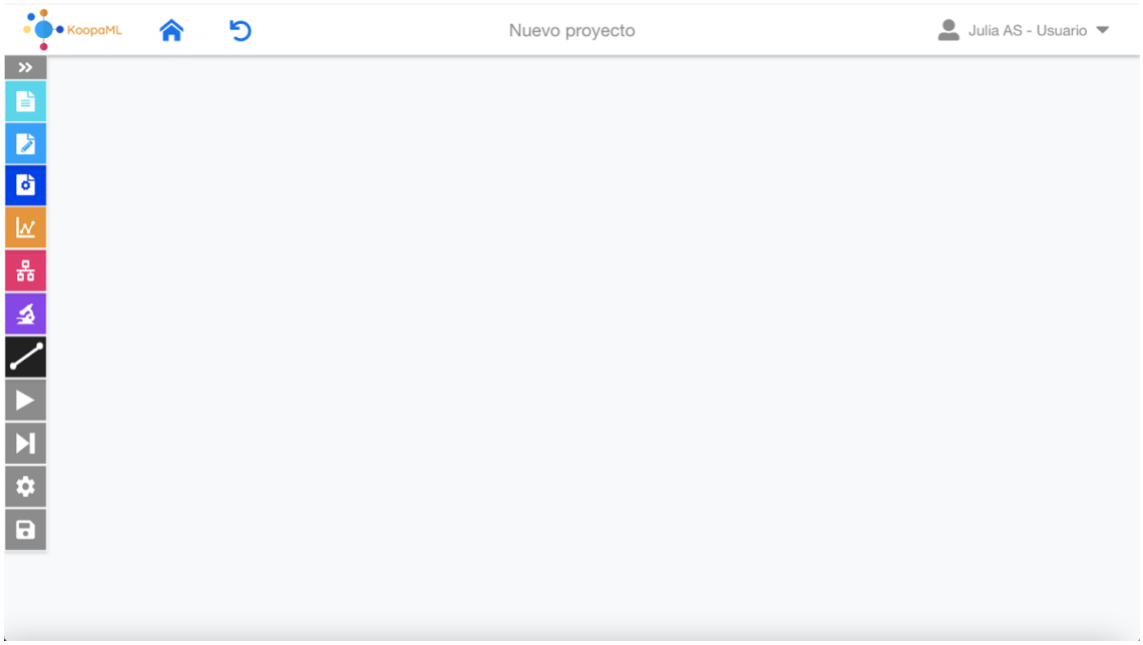


Figura 13. Pantalla nuevo proyecto prototipo inicial

- Segundo prototipo (después de evaluación):

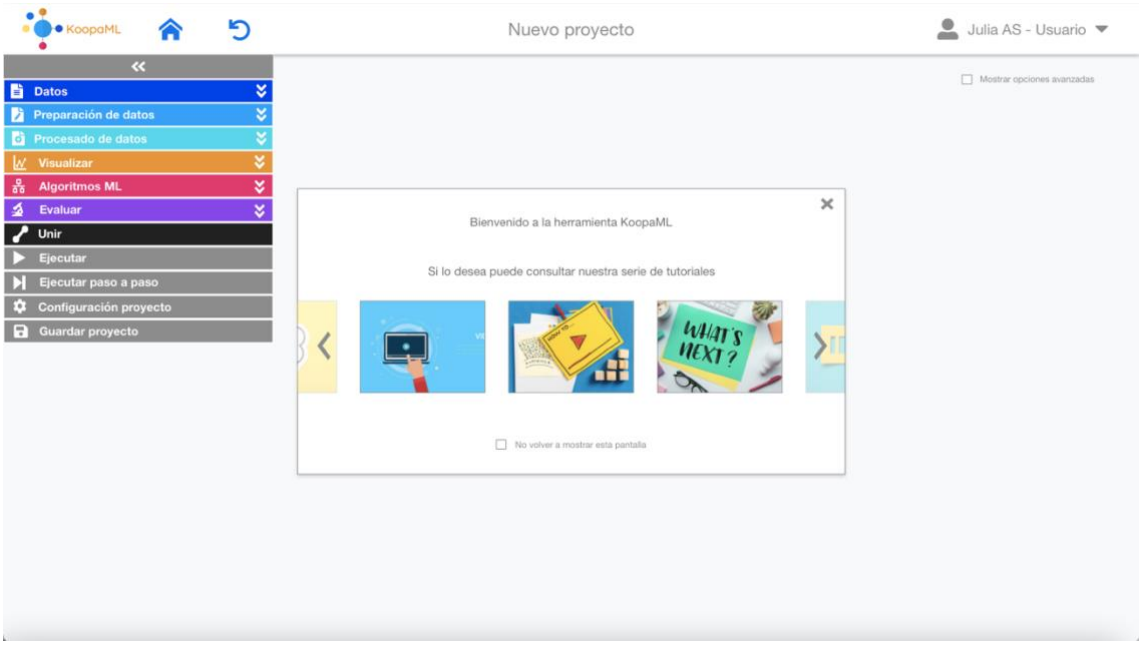


Figura 14. Pantalla nuevo proyecto prototipo definitivo

Como puede observarse en la Figura 14, el prototipo digital realizado tras la evaluación con los usuarios muestra la barra lateral del menú expandida, así como una ventana modal con varios tutoriales que cubrirían desde aspectos básicos del uso de la aplicación hasta otros más avanzados.

- Pantalla modificar proyecto:
  - Aspectos negativos:
    - "No queda claro qué tareas están en proceso de ejecución y cuáles han terminado de ejecutarse". (Acho, experto en ciencia de datos)
  - Sugerencias:
    - "Cada nodo al ejecutarse podría tener su "tick" para saber cuáles han sido ejecutados y cuáles aún no". (Antonio, experto en ciencia de datos)
  - Aspectos positivos:
    - "Fácil la forma de ejecutar y visualizar resultados" (Acho, experto en ciencia de datos)

- "Intuitiva la forma de llegar a la pantalla (los usuarios llegaron a esta pantalla rápidamente sin problemas)" (David, médico)

A continuación se muestran las imágenes de los dos prototipos digitales realizados, con el objetivo de ver su evolución tras la evaluación con el grupo focal de usuarios.

- Primer prototipo (antes de evaluación) ilustrado con la Figura 15.

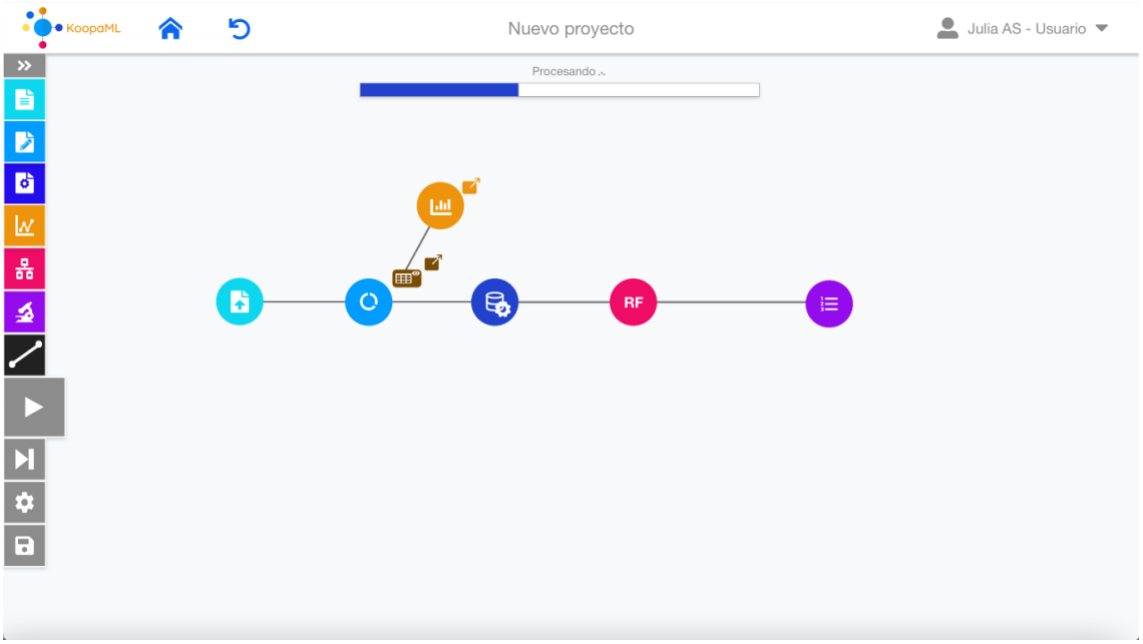


Figura 15. Pantalla modificar proyecto prototipo inicial

- Segundo prototipo (después de evaluación):

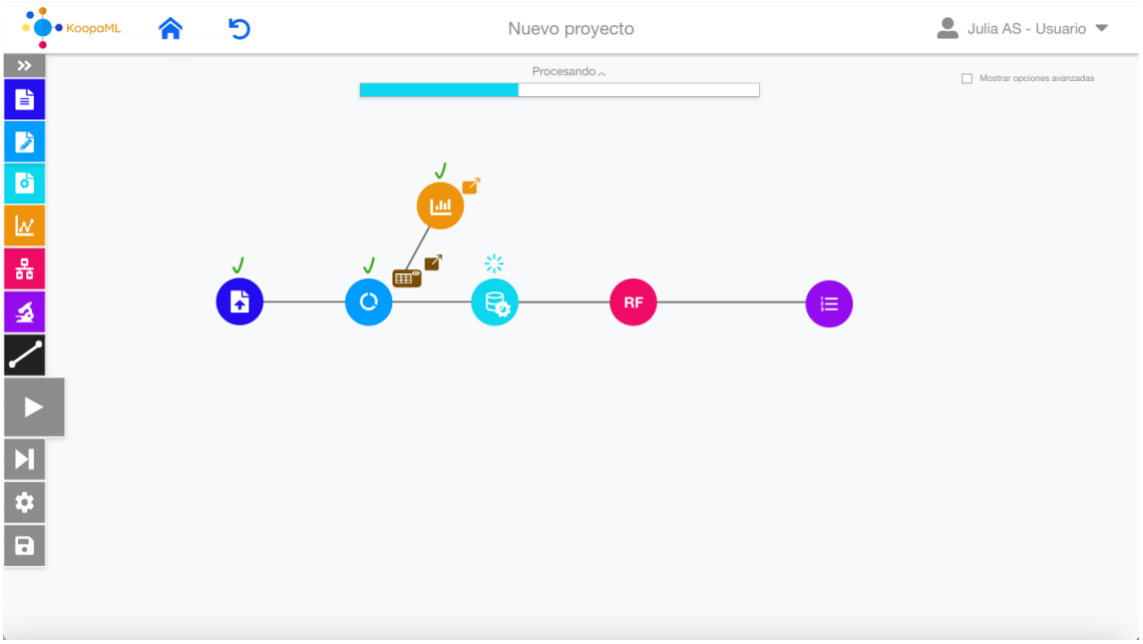


Figura 16. Pantalla modificar proyecto prototipo definitivo

Como puede observarse en la Figura 16, el prototipo digital realizado tras la evaluación con los usuarios muestra un icono en forma de tick sobre aquellos nodos que ya han sido ejecutados, así como un icono en forma de círculo emulando la animación de "cargando" en el nodo que se está ejecutando en ese momento.

- Todas las pantallas:
    - Aspectos positivos
        - "Sencillez de la herramienta general" (David, Eduardo, médicos)
        - "Herramienta bastante escalable" (Jesús, experto en ciencia de datos)
        - "La interfaz gusta mucho" (David, médico; Jesús, experto en ciencia de datos)
- Otro cambio posterior a este segundo prototipo (se puede observar en la aplicación real) es el cambio de forma de los nodos, pasando de ser un círculo a un cuadrado, pues se necesita ver con claridad las uniones y los parámetros de entrada y salida de los nodos y en un nodo circular no se podría apreciar claramente.
- Otro cambio también posterior a este segundo prototipo (también se puede observar en la aplicación real) es la eliminación de la opción de unir en el menú lateral, pues en la aplicación real las uniones se realizan haciendo clic sobre las entradas o salidas de lo nodos y dibujando una línea, por lo que esa opción carecía de utilidad en el menú.

5.3. Ingeniería del software

El diseño centrado en el usuario se ha combinado con la documentación de ingeniería necesaria para llevar a cabo la fase de desarrollo.

En primer lugar, la elicitación de requisitos se ha desarrollado a través del diseño centrado en el usuario y se ha documentado a través de casos de uso, esta tarea se desarrollo durante el tercer Sprint (semana 3). Cabe destacar que se han identificado

cuatro actores diferentes y se ha dividido la funcionalidad de la aplicación en cuatro grandes bloques:

- Diagramas de casos de uso bloque gestión usuarios

  Contiene la representación de los casos de uso relacionados con las altas, bajas y modificaciones de las cuentas de usuarios.
- Diagrama de casos de uso bloque gestión proyectos

  Contiene la representación de los casos de uso relacionados con la creación, descarga y eliminación de proyectos.
- Diagrama de casos de uso bloque gestión pipeline

  Contiene la representación de los casos de uso relacionados con la creación, eliminación, configuración, unión y ejecución de nodos.
- Diagrama de casos de uso bloque gestión heurísticas

  Contiene la representación de los casos de uso relacionados con la modificación de las heurísticas de la aplicación.

Se pueden consultar los diversos diagramas en el “Anexo 2. Proceso de ingeniería”.

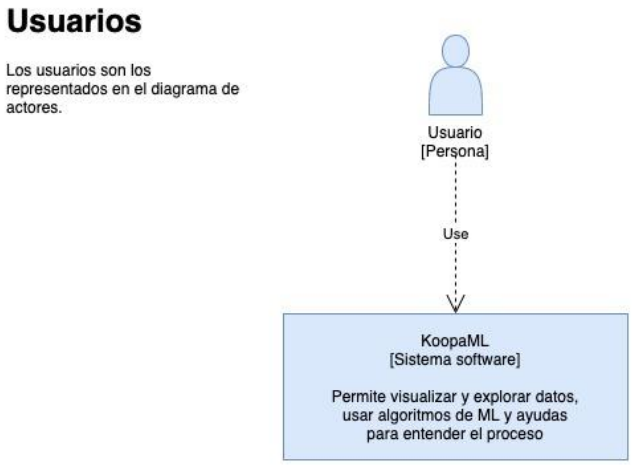
Por otro lado, desde el punto de vista de la ingeniería tiene especial relevancia la arquitectura del sistema. Para modelarla se ha utilizado un modelo de arquitectura C4, este modelo representa a través de varios niveles de abstracción la estructura y comunicación del sistema con los usuarios, esta tarea se ha desarrollado durante el quinto S print (semana 5). Para este sistema se han realizado modelos de los dos primeros niveles, pueden consultarse con más detalle en el Anexo 2. Los dos niveles representados corresponden a:

- Modelo de contexto:

  En este nivel se representa el sistema objetivo y la interacción de este con las personas u otros sistemas (mro, 2020)

  KoopaML es un sistema que será utilizado por usuarios (indicados en el diagrama de actores del anexo “Anexo 2. Proceso de ingeniería”) y les permitirá visualizar y explorar datos, entrenar algoritmos de Machine Learning y aprender sobre este proceso.

  Este modelo se muestra en la Figura 17.



.

Figura 17. Modelo arquitectura C4 contexto

- Modelo de contenedor

  En este nivel disminuye el grado de abstracción, pues se hace referencia a aplicaciones, almacenamiento de datos, micro servicios…

  El modelo de contenedor de KoopaML ilustra cómo el usuario utiliza su navegador para comunicarse con el sistema; el navegador (frontend) es quien se comunica con el servidor (backend) y la API. A su vez es el servidor (backend) quien se comunica con la base de datos no relacional. El modelo de contenedor se muestra en la Figura 18.

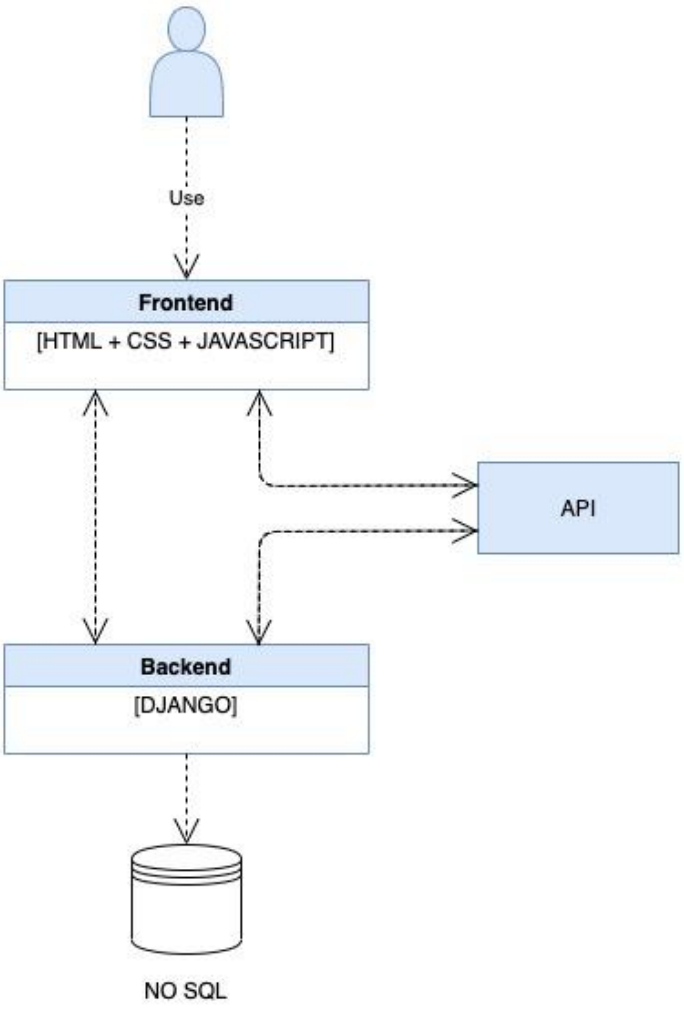


Figura 18. Modelo arquitectura C4 contenedor

5.4. Almacenamiento en base de datos

Tal y como se explicó en el Capítulo 3. Técnicas y herramientas, se ha utilizado una base de datos no relacional para el almacenamiento de la información, debido a la gran variedad de tipos de datos existentes en la aplicación.

Este aspecto supuso un desafío personal, pues no se había trabajado antes con este tipo de bases de datos.

Para que la aplicación sea extensible y pueda adaptarse a cualquier tipo de sistema se ha decidido estructurar la base de datos de la siguiente forma: el sistema presenta tres entidades, la entidad correspondiente a los usuarios es Usuario, la que corresponde al flujo de nodos o pipeline es Flujo y la que corresponde a cada nodo se determina Nodo. La entidad Usuario almacena atributos como el nombre único de usuario, el email, el nombre, los apellidos, la contraseña, el rol y si el usuario se encuentra o no validado.

La entidad Flujo (pipeline) almacena como atributos el identificador único de flujo, el usuario autor de ese flujo, el nombre del flujo y el flujo de nodos en formato JSON.
La entidad Nodo almacena atributos como el identificador único de cada nodo, el flujo o pipeline al que pertenece, el tipo de nodo, el identificador de ese nodo en el flujo, las entradas del nodo, las salidas del nodo, los parámetros configurados para ese nodo, los resultados intermedios que genera la ejecución de ese nodo y la fecha de creación del nodo. Dentro de los atributos de estas entidades se encuentran algunos cuyo tipo no puede definirse de forma estática, pues cada tipo de nodo almacenará un tipo de dato diferente en cada campo; este es el caso de los campos “resultados” y “parámetros” tal y como se explicó en el Capítulo 3, que se guardarán en forma de diccionario en formato JSON.
Las relaciones existentes son las siguientes:

- Un usuario puede ser el autor de 0 o más flujos o pipelines, pero un flujo sólo puede pertenecer a un usuario.
- Un flujo contiene 0 o más nodos, pero un nodo sólo puede pertenecer a un flujo o pipeline.

A continuación se muestra a través de la Figura 19 el diagrama que ilustra esta explicación:

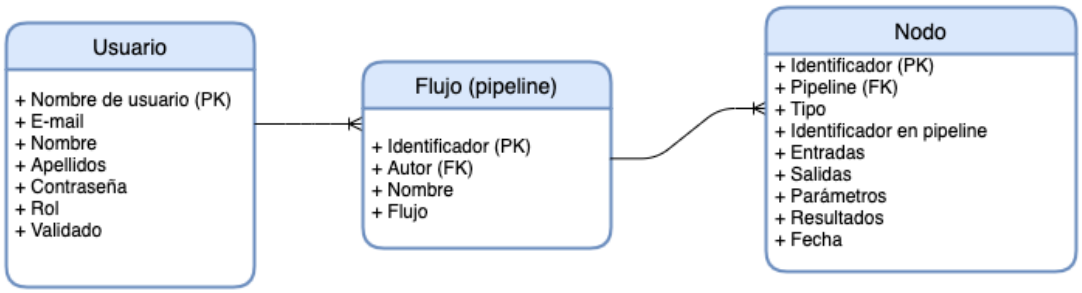


Figura 19. Diagrama entidad-relación

De esta forma se facilita la tarea de añadir nuevas actividades o nodos a la aplicación, pues únicamente se ha de crear una nueva instancia de la entidad Nodo y rellenar los campos correspondientes con los datos del nodo en cuestión, sin necesidad de añadir o eliminar entidades de la base de datos. Lo mismo ocurre con los usuarios y los flujos o pipelines.

5.5. Utilización y adaptación de bibliotecas

Cabe destacar que, aunque se haya hecho uso de librerías ya existentes, se ha dedicado un gran trabajo en la adaptación de estas para una correcta integración en los objetivos de la aplicación.

5.5.1. Rete.js

Se ha hecho uso de la biblioteca rete.js para la creación y modificación de los nodos presentes en la aplicación.

Rete.js es un framework modular de JavaScript orientado a la programación visual (Rete.js, s.f.).

Se ha adaptado para la utilización en este sistema definiendo cada nodo o actividad como un componente acompañado de un control.

Los nodos del framework inicialmente mostraban la estructura que se observa en la Figura 20.

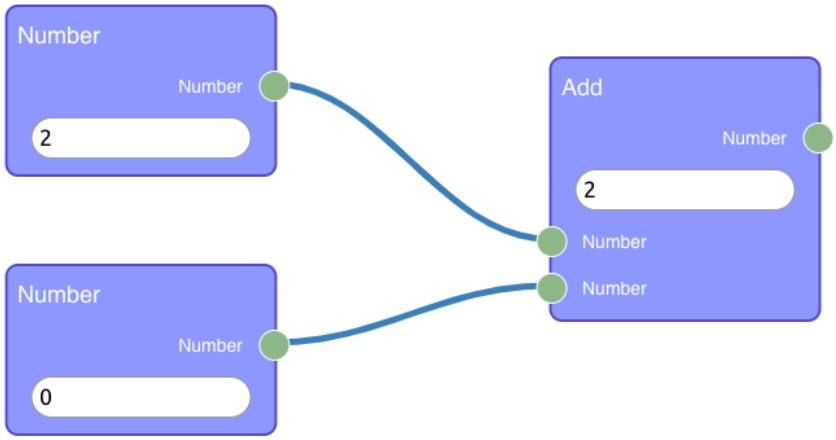


Figura 20. Framework rete.js inicial (Rete.js, s.f.)

Para esta aplicación, como se explicó en párrafos anteriores, se ha utilizado un nodo para representar una tarea o actividad, por lo que las entradas, salidas y parámetros de los nodos varían en función de la actividad que represente, tal y cómo puede observarse en la Figura 21, así como el estilo de cada nodo.

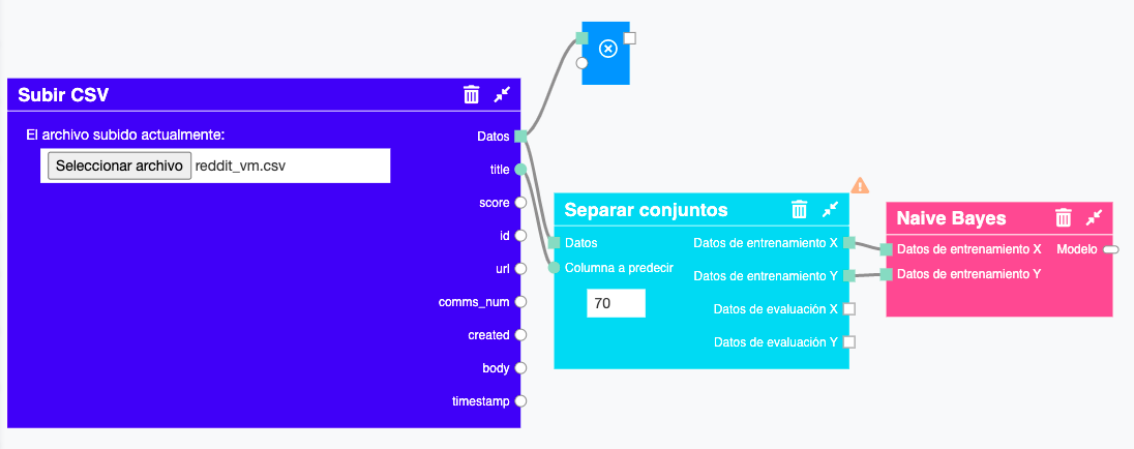


Figura 21. Framework rete.js personalizado

Se ha añadido funcionalidad extra a los nodos, de forma que algunos de ellos, como es el caso del nodo "Subir CSV" que se muestra en la figura 20, modifican sus variables de salida en función del archivo que el usuario introduce como parámetro en tiempo real, analizando el fichero subido y recuperando las variables.

Otra funcionalidad añadida es la actualización en tiempo real del flujo de nodos en la base de datos, es decir, cada vez que se realiza una modificación en el lienzo, en cualquiera de los nodos, el flujo de nodos se actualiza en la base de datos, así como el nodo o nodos afectados. Este aspecto se trata con más detalle en el apartado 5.6. Actualización en tiempo real.

Otro ejemplo de modificación realizada para que encajara mejor con el objetivo de la aplicación es que los nodos, tal como puede observarse en la Figura 20, pueden expandirse y contraerse, de forma que la interfaz resulta más clara y sencilla.

Además se han realizado modificaciones en el estilo de los nodos, reflejadas en el apartado 5.1. Diseño centrado en el usuario, para lograr una interfaz clara que cumpliera todos los requisitos de los usuarios. Entre estas modificaciones estéticas está la existencia de un botón de eliminar en todos los nodos, reemplazando el menú contextual que se ofrecía por defecto en la biblioteca; otra modificación estética es la variabilidad del color del nodo y el icono que ilustra su comportamiento en función de la actividad que representa, así como las diferentes formas de los sockets representantes de las conexiones según el tipo de dato que ofrecen.

Otro cambio realizado para adaptar la biblioteca a los requisitos solicitados es la forma en la que se ha modelado la incorporación de los nodos al lienzo. Aunque esta biblioteca

facilitaba un menú que permitía arrastrar nodos para añadirlos al lienzo tenía un aspecto que no podía ser personalizado y no encajaba con los requisitos de diseño expuestos en fases anteriores, pues suponía una sobrecarga de información en la interfaz, además de no permitir flexibilidad ni escalabilidad, tal y como se puede observar en la Figura 22.

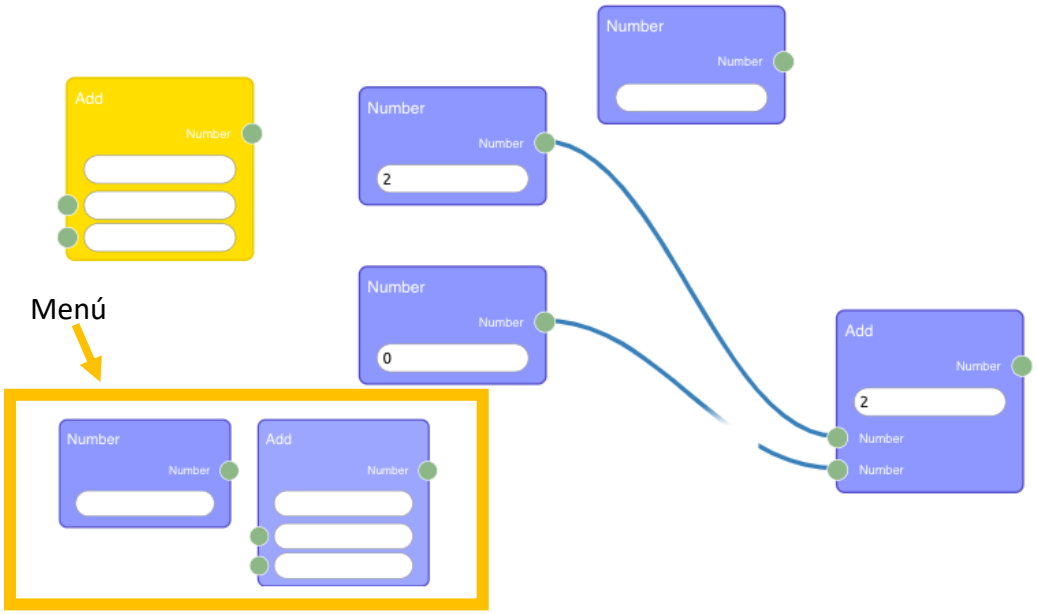


Figura 22. Menú arrastrar y soltar ofrecido por Rete.js

A continuación se muestran los nodos creados, con una breve explicación de su funcionamiento.

El primer nodo pertenece a la categoría de Datos y se denomina “Subir CSV”, está representado por el bloque mostrado en la Figura 23. Su función es permitir al usuario subir un archivo .csv a la aplicación, cuyo contenido podrá ser utilizado por otros nodos para diversas tareas. Las salidas de este nodo son dinámicas, es decir, cuando el usuario introduce un archivo el nodo actualiza las variables de salida en función del contenido del archivo, como puede observarse en la Figura 21.

Figura 23. Nodo Subir CSV

El segundo nodo pertenece a la categoría de Preparación de datos y se denomina "Valores perdidos", está representado por el bloque mostrado en la Figura 24. Su función es mostrar si existen valores perdidos en los datos introducidos como entrada, este nodo ofrece la posibilidad de realizar la búsqueda de valores perdidos sobre todos los datos o sobre una o varias columnas en específico (entrada "Variables"). Este nodo, una vez ejecutado el flujo al que pertenece, ofrece al usuario la posibilidad de revisar su resultado en forma de tabla.

Figura 24. Nodo Valores perdidos

El tercer nodo pertenece a la categoría de Procesado de datos y se denomina "Separar conjuntos", está representado por el bloque mostrado en la Figura 25. Su función es separar los datos en cuatro conjuntos para el entrenamiento y la evaluación de los modelos. Este nodo, al igual que el anterior, ofrece al usuario la posibilidad de revisar el resultado de la separación en formato de tabla.

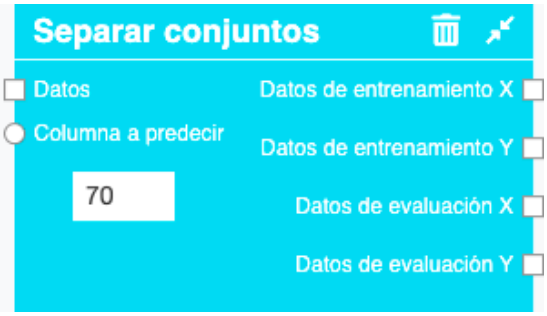


Figura 25. Nodo Separar conjuntos

El cuarto nodo pertenece a la categoría de Procesado de datos y se denomina "Rellenar valores perdidos", está representado por el bloque mostrado en la Figura 26. Su función es rellenar los valores perdidos que se encuentren en los datos introducidos, pudiendo escoger, al igual que en el segundo nodo, si rellenar los valores perdidos de todos los datos o únicamente de algunas columnas. Además, este nodo tiene un parámetro adicional que permite escoger si rellenar los valores perdidos con el valor de la media o

con el valor más frecuente entre los datos. Este nodo, al igual que los anteriores, ofrece al usuario la posibilidad de revisar el resultado de la separación en formato de tabla.

Figura 26. Nodo Rellenar valores perdidos

El quinto nodo pertenece a la categoría de Algoritmos de aprendizaje automático y se denomina “Naive Bayes”, está representado por el bloque mostrado en la Figura 27. Su función es entrenar un modelo de Naive Bayes. Su salida es el modelo entrenado, y tras la ejecución del flujo al que pertenece este nodo este modelo puede ser descargado.

Figura 27. Nodo Naive Bayes

El sexto nodo pertenece a la categoría de Algoritmos de aprendizaje automático y se denomina “Random Forest”, está representado por el bloque mostrado en la Figura 28. Su función es entrenar un modelo de Random Forest. Su salida es el modelo entrenado, y tras la ejecución del flujo al que pertenece este nodo el modelo puede ser descargado.

Figura 28. Nodo Random Forest

El séptimo nodo pertenece a la categoría de Algoritmos de aprendizaje automático y se denomina “Support Vector Machine”, está representado por el bloque mostrado en la Figura 29. Su función es entrenar un modelo de Support Vector Machine. Su salida es el modelo entrenado, y tras la ejecución del flujo al que pertenece este nodo el modelo puede ser descargado.

Figura 29. Nodo Support Vector Machine

El octavo nodo pertenece a la categoría de Algoritmos de aprendizaje automático y se denomina “Linear Regression”, está representado por el bloque mostrado en la Figura 30. Su función es entrenar un modelo de Linear Regression. Su salida es el modelo entrenado, y tras la ejecución del flujo al que pertenece este nodo el modelo puede ser descargado.

Figura 30. Nodo Linear Regression

El noveno nodo pertenece a la categoría de Evaluación y se denomina “Precisión”, está representado por el bloque mostrado en la Figura 31. Su función es comprobar la precisión de un modelo entrenado. El resultado proporcionado por este nodo puede ser visualizado tras la ejecución.

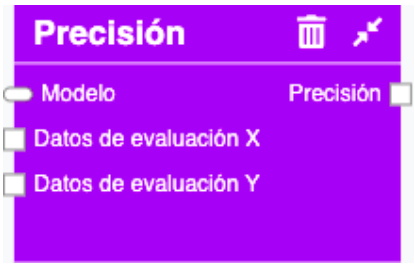


Figura 31. Nodo Precisión

Cabe destacar que, aunque por el momento sólo existen estos nodos la estructura genérica, que soporta la creación y actualización de nodos ya está creada, minimizando enormemente el esfuerzo que supondría la acción de añadir un nuevo nodo.

5.5.2. FlowChart.js

Se ha hecho uso de la biblioteca FlowChart.js para la representación en forma de diagrama de flujo de las heurísticas definidas en la aplicación. De forma que los usuarios con el rol de administrador o experto puedan codificar heurísticas en un lenguaje similar al pseudocódigo, como se muestra en la Figura 32, y visualizarlas de manera gráfica, tal y como se muestra, en parte, en la Figura 33.

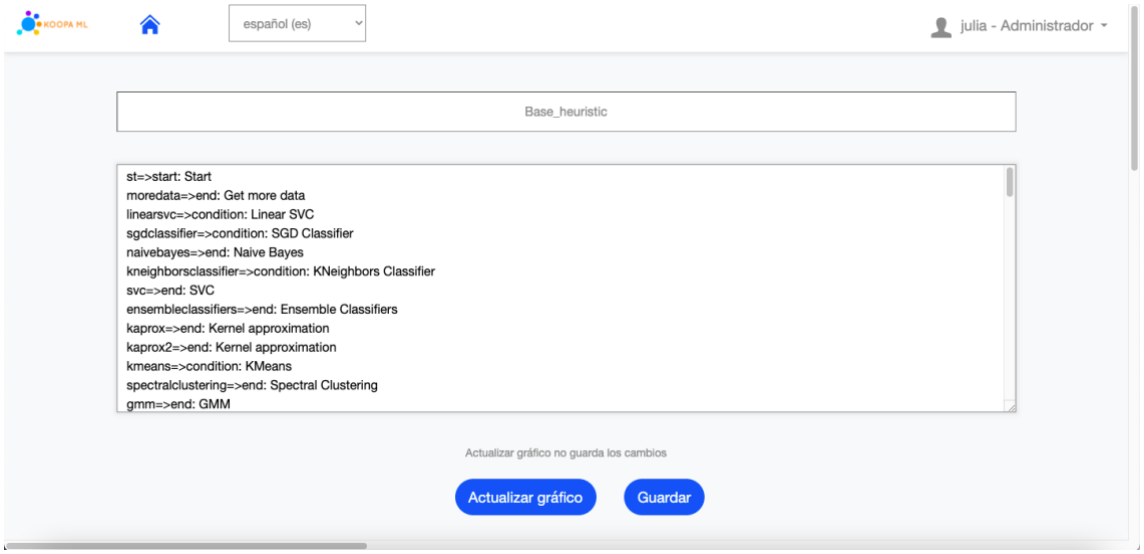


Figura 32. Pseudocódigo heurística base

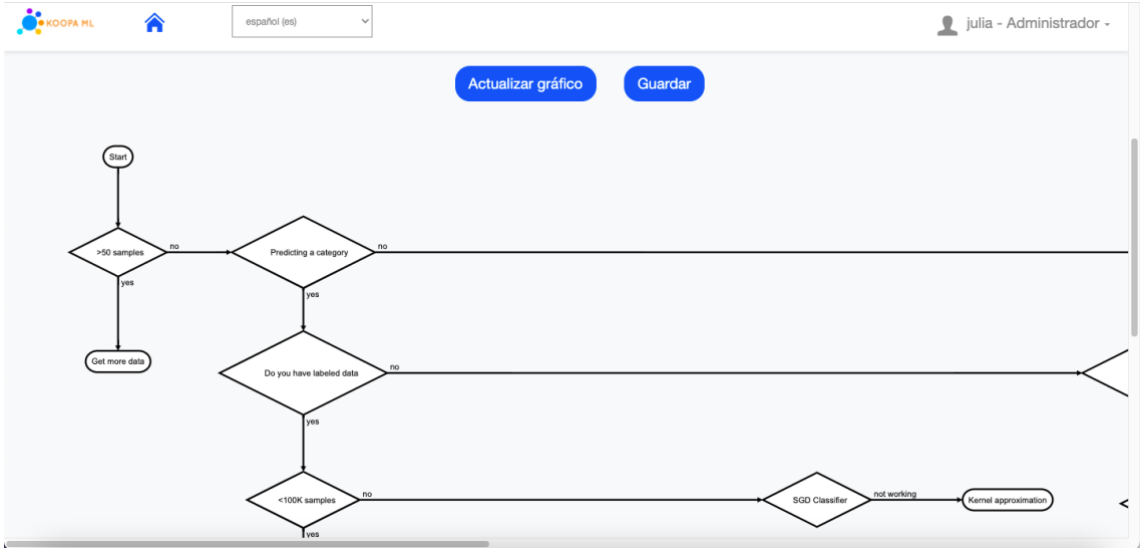


Figura 33. Parte del diagrama de flujo correspondiente a la heurística base

La aplicación permite la creación, modificación y eliminación de heurísticas, a excepción de la heurística base, que no puede ser modificada ni eliminada.

Estas heurísticas servirán en un futuro como fuente de sugerencias para guiar al usuario sobre los mejores algoritmos a aplicar según los datos introducidos, la tarea que se desee realizar, etc.

Para la codificación de la heurística base se ha utilizado el árbol de decisión de scikit learn (scikit learn, s.f.).

5.6. Actualización del flujo en tiempo real

Los nodos pertenecientes a la funcionalidad de creación y edición de flujos de la aplicación tienen asociados iconos que ofrecen soporte en forma de información sobre errores, advertencias, sugerencias e indicaciones de visualizaciones o descargas disponibles.

El significado de los iconos utilizados es el siguiente:

- El icono indica que hay una visualización de resultados disponible.
- El icono indica que hay una descarga de resultados disponible.
- El icono indica que hay un error.
- El icono indica que hay una advertencia.
- El icono indica que hay una sugerencia.

Estos iconos pueden observarse en la Figura 34.

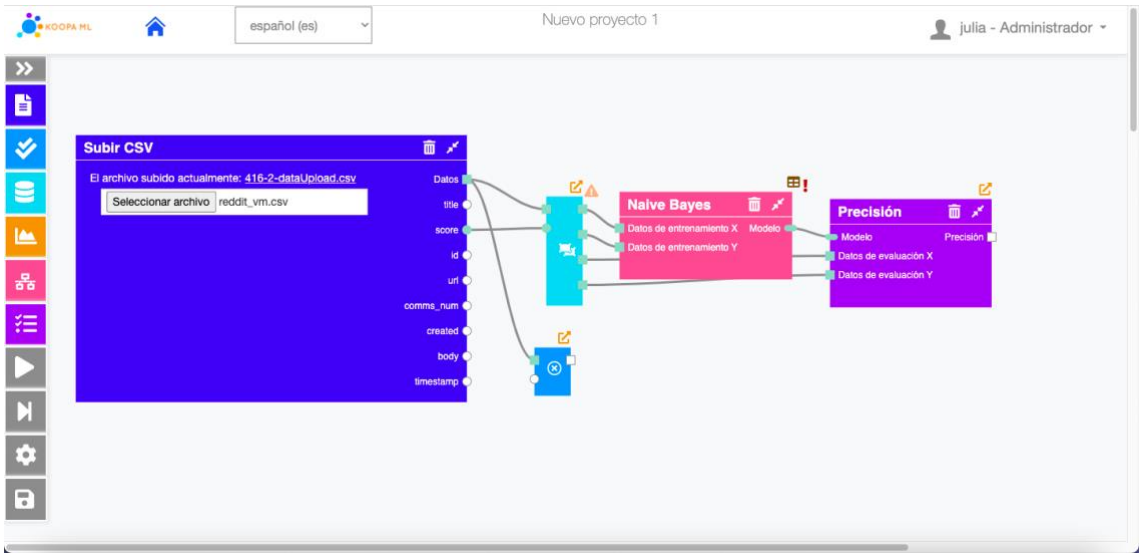


Figura 34. Iconos en tiempo real

Estos iconos deben mostrarse en tiempo real, pues cada cambio en un nodo puede provocar la aparición o desaparición de estos símbolos.

Está es la razón por la que se decidió realizar, tal y como se mencionó en el apartado anterior, actualizaciones del flujo en tiempo real, actualizando la instancia correspondiente en la base de datos cada vez que se modifica un nodo. Para lograr esta actualización dinámica se desarrolló una función denominada "functUpdate" que se ejecuta cada vez que se produce un cambio en algún nodo, como puede observarse en la Figura 35.

```
/*Comprobar datos perdidos*/
var VueMissingDataControl = {
    props: ['readonly', 'emitter', 'ikey', 'getData', 'putData'],
    template: '<@input="change($event)" @dblclick.stop=""/>',
    data() {
        return {
            value: null,
        }
    },
    methods: {
        change(e) {
            //Call pipeline update
            functUpdate().then(
                () => {
                    this.update();
                })
        },
```

Figura 35. Ejemplo llamada a función actualización flujo

Esta función, tal como puede observarse en la Figura 36 realiza una llamada a través de Ajax a una vista definida como "update/", indicándole el identificador único del flujo a actualizar y pasándole como datos el flujo de nodos actual en formato JSON.

```
function functUpdate() {
    console.log(window.editor.toJSON());
    var data = JSON.stringify(window.editor.toJSON());
    console.log(data);
    return $.ajax({
        url: '/update/' + Number.parseInt(id_pipeline),
        type: 'POST',
        data: data,
        processData: false,
        contentType: false,
        beforeSend: function (xhr, settings) {
            xhr.setRequestHeader("X-CSRFToken", csrf_token);
        },
        success: function (data) {
            console.log(data);
        },
        error: function () {
            console.log("Error");
        }
    });
}
```

.

Figura 36. Función functUpdate() JavaScript

La vista en cuestión se presenta en la Figura 37. Esta vista recupera el flujo de nodos correspondiente almacenado en la base de datos a través del identificador que se le pasa como parámetro. Tras esto actualiza el campo “flow” (flujo) a través de la información del campo “datos” de la llamada ajax. Recorre nodo a nodo este nuevo flujo, comprobando para cada nodo si este existe en la base de datos o no. En caso negativo crea una nueva instancia de nodo con los atributos correspondientes. En caso afirmativo actualiza la instancia correspondiente. Tras finalizar esta tarea realiza otra comprobación, para cada nodo almacenado en la base de datos perteneciente al flujo cuyo identificador se ha recibido se comprueba si existe en el flujo actual o no. En caso afirmativo no se realiza ninguna acción, en caso negativo se elimina el nodo y sus ficheros asociados (en caso de que el nodo tenga ficheros asociados).

```
@login_required()
def update(request, id_pipeline):
    owner_of_pipeline = Pipeline.objects.get(id=id_pipeline).user
    if request.user != owner_of_pipeline:
        return redirect('my_area')
    if request.method != 'POST':
        return HttpResponseBadRequest('Only POST requests are allowed')
    else:
        pipeline_to_update = Pipeline.objects.get(id=id_pipeline)
        pipeline_to_update.flow = json.loads(request.body)
        pipeline_to_update.save()
        nodes = pipeline_to_update.flow.get('nodes')
        for key in nodes:
            node = nodes[key]
            node_exists = Node.objects.filter(pipeline=pipeline_to_update,
number_in_pipeline=node.get('id')).exists()
            if not node_exists:
                print(node)
                print(node.get('data'))
                if node.get('name') == 'Subir CSV':
                    node_created =
Node.create(number_in_pipeline=node.get('id'),pipeline=pipeline_to_update,
node_type=node.get('name'),inputs=node.get('inputs'), outputs=node.get('outputs')
parameters={}, results={})
                else:
                    node_created=
Node.create(number_in_pipeline=node.get('id'),pipeline=pipeline_to_update,node_type=node.g
et('name'),inputs=node.get('inputs'),
outputs=node.get('outputs'),parameters=node.get('data'), results={})
            else:
                node_existing = Node.objects.get(number_in_pipeline=node.get('id'),
pipeline=pipeline_to_update)
                if node_existing.node_type == 'Subir CSV':
                    node_existing.update_node(id=node_existing.id,
pipeline=pipeline_to_update, node_type=node.get('name'), inputs=node.get('inputs'),
outputs=node.get('outputs'), parameters=node_existing.parameters,
results=node_existing.results)
                else:
                    node_existing.update_node(id=node_existing.id,
pipeline=pipeline_to_update,node_type=node.get('name'),inputs=node.get('inputs'),
outputs=node.get('outputs'), parameters=node.get('data'), results=node_existing.results)

        for node_saved in Node.objects.filter(pipeline__id=id_pipeline):
            node_to_delete = True
            for key in nodes:
                if node_saved.number_in_pipeline == nodes[key].get('id'):
                    node_to_delete = False
                    break
            if node_to_delete:
                if node_saved.node_type == 'Subir CSV':
                    if node_saved.parameters != {}:
                        path_file = json.loads(node_saved.parameters).get('file-path-
dataUpload')
                        if os.path.exists(path_file):
                            os.remove(path_file)
                else:
                    if node_saved.results != {}:
                        results = json.loads(node_saved.results)
                        for key in results:
                            print(key)
                            print(results[key])
                            path_file = results[key]
                            if os.path.exists(path_file):
                                os.remove(path_file)
                node_saved.delete()
        return HttpResponse('correcto')
```

Figura 37. Código vista update

De esta forma los errores, advertencias, sugerencias y vínculos se actualizan en el mismo instante que se produce un cambio. No obstante se ha de tener en cuenta que aunque ciertos errores, advertencias y sugerencias pueden ser mostrados en cualquier momento, como puede ser el error “No hay archivo de datos” del nodo Subir CSV, hay otros avisos que sólo se actualizan tras ejecutar el flujo, como pueden ser los correspondientes a las indicaciones de disponibilidad de visualización o descarga de resultados intermedios.

Otra ventaja de esta solución es que, al actualizarse la información del flujo en la base de datos con cada cambio, si el usuario cierra la pestaña de forma inintencionada el proyecto estará guardado, por lo que no habrá perdido sus progresos.

Cabe destacar que la actualización en tiempo real no supone ejecutar el código, únicamente se actualizan las instancias de los nodos en la base de datos y en la interfaz. El flujo sólo se ejecuta cuando el usuario presiona el botón correspondiente a la acción de ejecutar, hasta entonces los nodos únicamente almacenan los datos necesarios para su futura ejecución, con esta decisión se persigue el objetivo de aumentar el rendimiento y eficiencia del sistema.

5.7. Multilingüe

Puesto que uno de los objetivos de la aplicación es que sea escalable, se ha decidido que la aplicación permita al usuario elegir entre dos idiomas: español o inglés. No obstante, la implementación a través de la cual se ha llevado a cabo permite la traducción a los idiomas que se desee.

Para realizar estas traducciones en el fichero settings.py se han añadido las opciones que muestra la Figura 38.

```
LANGUAGE_CODE = 'es'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True

LANGUAGES = (
    ('en', 'English'),
    ('es','Español'),
)

LOCALE_PATHS = (
    os.path.join(BASE_DIR, 'locale'),
)
```

Figura 38. Fragmento código internacionalización Django

También se ha añadido al archivo urls.py de la aplicación la siguiente ruta:

```
path('i18n/', include('django.conf.urls.i18n')),
```

Para poder traducir textos se ha de proceder de diferente forma según este texto se halle en una vista, una plantilla o un script de JavaScript.

En el primer caso, en el que el texto a traducir se encuentre en una vista se utiliza django.utils.translation, importando gettext como _. De esta forma, anteponiendo el caracter _ a cualquier texto indicaremos que puede ser traducido, como puede observarse en la Figura 39.

```
context={"error": _('Este nombre de usuario ya está en uso'),
         "form": form})
```

Figura 39. Fragmento código internacionalización Django en vista

En el segundo caso, en el que el texto a traducir se encuentra en una plantilla, se ha de cargar al inicio “i18n”. Tras ello se podrán indicar los textos a traducir haciendo uso de la línea {% trans mensaje %} como se puede observar en la Figura 40.

```
<a class="dropdown-item"
   href="{% url 'logout' %}">{% trans 'Cerrar sesión' %}</a>
```

Figura 40. Fragmento código internacionalización Django en plantilla HTML

El tercer caso, en el que el texto a traducir se encuentra en JavaScript presenta mayor dificultad, pues no puede ser traducido importando ningún comando. Por ello es necesario crear en la plantilla HTML que se use con el script en cuestión una serie de

variables que contengan los textos, indicando que estos podrán ser traducidos (tal y como se muestra en la Figura 40). De esta forma en el script se hará referencia a la variable que contiene el texto. Esto puede observarse en las figuras Figura 41 y Figura 42.

```
let input1 = new Rete.Input('dataToFill', data, dataSocket);
let input2 = new Rete.Input('variablesToFill', variables_to_fill, varSocket, true);
let output1 = new Rete.Output('dataFilled', filled_data, dataSocket);
```

Figura 41. Fragmento código internacionalización Django JavaScript

Las variables “data”, “variables_to_fill” y “filled_data” corresponden al texto que se muestra en las entradas y salidas del nodo “Rellenar valores perdidos”. Estas variables, entre otras, están definidas en una plantilla HTML como se muestra en la Figura 42.

```
var tooltip_split_sets = '{% trans "Tamaño conjunto entrenamiento (1 a 100)" %}'
var upload_csv = "{% trans 'Subir CSV' %}"
var data = "{% trans 'Datos' %}"
var missing_values = "{% trans 'Valores perdidos' %}"
var variables = "{% trans 'Variables' %}"
var fill_data = "{% trans 'Rellenar valores perdidos' %}"
var variables_to_fill = "{% trans 'Variables a rellenar' %}"
var filled_data = "{% trans 'Datos rellenos' %}"
var split_sets = "{% trans 'Separar conjuntos' %}"
var col_to_predict = "{% trans 'Columna a predecir' %}"
var train_data_x = "{% trans 'Datos de entrenamiento X' %}"
var train_data_y = "{% trans 'Datos de entrenamiento Y' %}"
var test_data_x = "{% trans 'Datos de evaluación X' %}"
var test_data_y = "{% trans 'Datos de evaluación Y' %}"
var model = "{% trans 'Modelo' %}"
var accuracy = "{% trans 'Precisión' %}"
```

Figura 42. Fragmento código internacionalización Django JavaScript y HTML

Tras ello se genera una carpeta para cada idioma a través del comando:

```
$ django-admin.py makemessages -l zh
```

En la carpeta generada se encuentra un archivo denominado django.po donde se encuentran todas estas cadenas de texto en el lenguaje en el que fueron escritas, junto con una cadena vacía donde se ha de colocar la traducción.

Para que las traducciones estén disponibles al modificar el lenguaje en la interfaz ha de ejecutarse el comando:

```
$ django-admin.py compilemessages
```

5.8. Gestión de archivos

La funcionalidad de crear o editar flujos de nodos de la aplicación contiene un nodo denominado “Subir CSV” que permite a los usuarios subir sus propios archivos. Estos archivos deberán quedar almacenados en el servidor siendo referenciados de forma única y no permitiendo el acceso a ellos a usuarios que no sean los propietarios de este. Además, puesto que la biblioteca Rete.js no permite almacenar en formato JSON el archivo introducido como parámetro a este nodo será necesario buscar una solución para que los usuarios tengan conocimiento del archivo que había sido cargado en ese nodo cuando vuelvan a entrar al proyecto.

Así mismo otros nodos como “Separar conjuntos”, “Rellenar valores perdidos” o todos los correspondientes a los algoritmos de aprendizaje automático, generan resultados intermedios que deberán almacenarse en el servidor, pudiendo ser visualizados únicamente por el usuario propietario del flujo de nodos.

Es necesario pues implementar soluciones que permitan una correcta y eficiente gestión de estos archivos.

La solución implementada para que los archivos no sean accesibles por el resto de usuarios se basa en dos premisas, la primera es que los archivos, tanto los subidos por los usuarios como los generados como resultados intermedios, no son accesibles a través de la URL, pues no se encuentran en la carpeta “static” (Documentation, django, s.f.). La segunda es que antes del acceso a cualquier archivo se comprueba que el usuario que desea acceder a él es el propietario del flujo en el que se encuentra el archivo, no permitiendo el acceso si no fuera así.

Para solucionar el problema de la identificación única se ha recurrido a la construcción de identificadores irrepetibles de la siguiente manera: cuando un usuario sube un archivo este se almacena en una carpeta del servidor denominada “UploadFiles” con un identificador único conformado por el número de pipeline o flujo (este es un identificador único en todo el sistema) y el número del nodo al que está asociado este archivo (este identificador es único en cada flujo). Este identificador se almacena en el campo “parámetros” del nodo en cuestión. Cuando un usuario cambia el archivo asociado al nodo en cuestión, ese archivo se reemplaza en el servidor. Así mismo cuando

un usuario elimina un nodo que tenía un archivo asociado este archivo es eliminado del servidor.

Lo mismo ocurre con los archivos generados como resultados intermedios.

Cabe destacar que el almacenamiento de las referencias a estos archivos se ha de realizar siguiendo una sintaxis determinada para lograr una infraestructura genérica que soporte la incorporación de nuevos nodos con independencia de su tipo (para más información consultar apartado 5.11. Capacidad de evolución y extensión):

- Archivos subidos al servidor por el usuario: su referencia ha de almacenarse en el campo “parameters” del nodo en cuestión, en forma de diccionario con el siguiente formato:
    - {'file-path-dataUpload': <path>}

  Donde <path> es la ruta del archivo en cuestión, en formato:
    - path = ‘UploadFiles/<name>’

  Donde <name> es el identificador del archivo en formato:
    - name = <id_pipeline>-<id_node>-<output>.csv

  Donde <id_pipeline> es el identificador del flujo al que pertenece el nodo, <id_node> es el identificador del nodo al cual se asocia el archivo y <output> es el nombre de la salida a la que se asocia ese archivo.
- Archivos generados como resultados por los nodos: su referencia ha de almacenarse en el campo “results” del nodo en cuestión, en forma de diccionario con el siguiente formato:
    - {'file-path-<output>: <path>}

  Donde <output> es la salida a la cual se asocia el archivo resultado generado y <path> es la ruta del archivo en cuestión, en formato:
    - path = ‘ResultFiles/<name>’

  Donde <name> es el identificador del archivo en formato:
    - name = <id_pipeline>-<id_node>-<output>.<extensión>

  Donde <id_pipeline> es el identificador del flujo al que pertenece el nodo, <id_node> es el identificador del nodo al cual se asocia el archivo, <output> es el nombre de la salida a la que se asocia ese archivo y <extensión> es la extensión del archivo en custión, ya sea .csv o .pkl en el caso de los modelos entrenados.

La solución al problema sobre cómo referenciar el archivo anteriormente cargado en el nodo “Subir CSV”, se basa en comprobar al abrir un proyecto ya creado si los nodos “Subir CSV” de ese flujo contienen en su campo parámetros el nombre de algún archivo. En caso afirmativo se recuperan y asocian con cada nodo correspondiente, tras esto la aplicación informa al usuario de cuál era el archivo que se había cargado en ese nodo, ofreciéndole un enlace para su descarga, como puede observarse en la Figura 43.

Figura 43. Nodo Subir CSV indicando archivo anteriormente cargado

5.9. Gestión de plantillas HTML y Bootstrap

Otro aspecto relevante es la gestión de las plantillas HTML y la incorporación de la biblioteca multiplataforma Bootstrap para conseguir un diseño web adaptativo.

El desarrollo de la parte frontend de la aplicación se ha llevado a cabo, como se mencionó en el Capítulo 3. Técnicas y herramientas con HTML, CSS y JavaScript. Este apartado se enfocará en la gestión de las plantillas utilizadas para facilitar la construcción de sitios web consistentes.

La biblioteca Bootstrap ha sido utilizado a lo largo de todo el desarrollo de frontend, con el objetivo de lograr un diseño adaptativo. Para ello se han utilizado los estilos y componentes que esta biblioteca ofrece, como son las filas, columnas, tablas, etc. Aunque los estilos de algunos de estos componentes han sido modificados para encajar en los requisitos del diseño de la aplicación.

Antes de comenzar con la explicación de la gestión de plantillas HTML implementada cabe destacar la división del proyecto en tres grandes partes:

- Gestión de usuarios
- Gestión de flujos (pipeline)
- Gestión de heurísticas

El bloque de gestión de flujos engloba los dos bloques funcionales: gestión de proyectos y gestión de flujo o pipeline definidos en el apartado 5.3. Ingeniería del software. Para cada uno de estos bloques se han realizado las plantillas necesarias. A continuación se presentarán las plantillas realizadas clasificadas por áreas:

- Plantillas gestión usuarios (UserManagment):
    - account
        - account_changes_admin.html: muestra al administrador una pantalla con los datos del usuario correspondiente y la posibilidad de editarlos.
        - account_changes_user.html: muestra al usuario una pantalla con sus datos personales y la posibilidad de modificarlos.
        - account_consult.html: muestra al usuario una pantalla con sus datos personales.
        - account_consult_all_users.html: muestra al administrador una pantalla con los datos de todos los usuarios de la aplicación.
        - adduser.html: muestra al administrador una pantalla con los datos a rellenar para añadir un nuevo usuario.
        - delete.html: muestra al administrador una pantalla solicitando confirmación para eliminar la cuenta de otro usuario.
        - delete_admin_confirm.html: muestra al administrador la pantalla de confirmación de la eliminación de la cuenta de un usuario.
        - delete_me.html: muestra al usuario una pantalla solicitando confirmación para la eliminación de su cuenta.
        - password_changes.html: muestra al usuario una pantalla a través de la cual cambiar su contraseña.
    - base
        - base.html: utilizada como plantilla base en las plantillas que se muestran cuando el usuario aún no ha iniciado sesión.
        - base2.html: utilizada como plantilla base en las plantillas que se muestran cuando el usuario ya ha iniciado sesión.
    - koopaMLApp

        - index.html: muestra la pantalla inicial de la aplicación.
    - my_area
        - delete_pipeline.html: muestra al usuario una pantalla pidiendo confirmación para la eliminación de un proyecto.
        - my_area_admin.html: muestra al administrador una pantalla con su área personal.
        - my_area_expert.html: muestra al experto una pantalla con su área personal.
        - my_area_user.html: muestra al usuario una pantalla con su área personal.
    - registration
        - login.html: muestra una pantalla solicitando los datos para iniciar sesión.
        - password_reset_complete.html: muestra al usuario una pantalla indicando que el cambio de contraseña se ha realizado correctamente.
        - password_reset_confirm.html: muestra una pantalla en la que se solicita la introducción de una nueva contraseña.
        - password_reset_done.html: muestra una pantalla en la que se indica que las instrucciones para el cambio de contraseña han sido enviadas al correo proporcionado.
        - password_reset_email.html: muestra el contenido del email que se envía al usuario para recuperar la contraseña.
        - password_reset_form.html: muestra la pantalla donde el usuario ha de introducir el correo al que se el enviarán las instrucciones para recuperar la contraseña.
        - password_reset_subject.txt
        - registry.html: muestra la pantalla de registro de un nuevo usuario.
        - registry_complete.html: muestra una pantalla que indica al usuario que su registro se ha realizado correctamente y ha de esperar a que el administrador valide su cuenta.

- Plantillas gestión flujo (pipelineManagement):
    - base
        - base3.html: utilizada como plantilla base en las plantillas que se muestran cuando el usuario accede a la creación o modificación de nodos.
        - base4.html: utilizada como plantilla base en las plantillas que se muestran con la visualización de los resultados intermedios.
    - canvas
        - canvas.html: muestra el lienzo sobre el que se construyen los flujos de nodos, así como el menú lateral donde se muestran los nodos a añadir.
    - results
        - display_table.html: muestra los resultados intermedios en formato de tabla.
- Plantillas gestión heurísticas (heuristicManagement):
    - base
        - base5.html: utilizada como plantilla base en las plantillas que se muestran cuando el usuario edita las heurísticas
    - diagram
        - heuristic_canvas.html: muestra la pantalla de edición de heurísticas.
        - heuristic_create.html: muestra la pantalla de creación de una nueva heurística.

5.10. Gestión de usuarios

Un aspecto importante y necesario de la aplicación es la gestión de los usuarios del sistema.

En primer lugar los usuarios pueden acceder a las funcionalidades de la aplicación a través del inicio de sesión, para ello necesitan ingresar su nombre de usuario o email y su contraseña.

Si un usuario no tiene aún cuenta de usuario puede registrarse introduciendo los datos requeridos, pero no podrá iniciar sesión hasta que el administrador valide su cuenta, evitando así que personal no autorizado tenga acceso al sistema.

Cualquier usuario registrado puede modificar sus datos personales en cualquier momento, así como modificar su contraseña. Así mismo si un usuario olvida su contraseña la aplicación implementa un mecanismo a través del cual se le solicita el email con el que está dado de alta, al cual se le enviarán automáticamente las instrucciones para la recuperación de la contraseña, como puede observarse en la Figura 44.

Figura 44. Correo recuperación contraseña KoopaML

Cabe destacar que cada usuario tiene un rol asociado que delimitará las acciones que puede realizar en la plataforma. De esta manera se distinguen tres roles:

- Los usuarios, generalmente médicos con poco conocimiento de Machine Learning, podrán acceder a la funcionalidad de crear y editar flujos de nodos, subir proyectos y modificar su cuenta personal, pero no podrán acceder a la funcionalidad de gestionar heurísticas, así como tampoco tendrán acceso a la información de las cuentas de otros usuarios.
- Los expertos, generalmente expertos en ciencia de datos, podrán acceder a las funcionalidades a las que tienen acceso los usuarios y además podrán modificar heurísticas.
- Los administradores, el rol con mayor libertad, podrán realizar todas las acciones que tienen permitidas los expertos y además tienen acceso a la información del resto de cuentas de usuarios, pudiendo modificarlas, eliminarlas o validarlas.

Por defecto al registrarse una cuenta nueva se les asocia el rol de usuario, pudiendo ser modificado este rol solo por el administrador.

5.11. Capacidad de evolución y extensión

Se ha de tener en cuenta que las disciplinas entre las que se enmarca esta aplicación están en constante evolución, por lo que el sistema ha de poder adaptarse a las necesidades cambiantes de los usuarios.

Esto se traduce en la necesidad de desarrollar una aplicación que permita la modificación del comportamiento de los nodos ya existentes, así como la incorporación de nuevos nodos en cualquiera de las categorías, e incluso la incorporación de nuevas categorías.

Por esta razón se ha desarrollado el sistema KoopaML de forma modular, se ha desarrollado la estructura interna de tal manera que soporte con facilidad la modificación de cada nodo y categoría, así como la incorporación de nuevas tareas.

Esto se ha logrado gracias a la implementación y desarrollo de cada nodo como un bloque completamente independiente del resto, el cual contiene en sus atributos toda la información necesaria para ser ejecutado, además el componente de nodo posee un control propio que gestiona su comportamiento con total independencia del resto.

Por otro lado la infraestructura que soporta la funcionalidad común de los nodos, como es el comportamiento del nodo al arrastrar y soltar, la posibilidad de contraerse o expandirse, la actualización del nodo en la base de datos (la cual puede observarse en la Figura 36) y la eliminación del nodo se ha desarrollado de forma que pueda ser utilizada de manera genérica con cualquier nodo que se añada al sistema, sin necesidad de aplicar cambio alguno a ninguna de estas funciones.

Sin embargo, para que la infraestructura genérica funcione correctamente con independencia del tipo de nodo se han de seguir las convenciones sintácticas descritas en el apartado 5.8. Gestión de archivos para aquellos nodos que trabajen con ficheros o generen resultados intermedios.

Por lo que si se desea añadir un nodo nuevo a la plataforma únicamente habría que gestionar algunos detalles estéticos, como el icono que lo representará, crear su componente y control y añadir la funcionalidad a la vista “run”, la cual se encarga de la ejecución de la funcionalidad única asociada a cada nodo del flujo, como puede observarse en el fragmento de código que muestra la Figura 45.

```
# ejecuto el run

for category in node_order:
    for node in Node.objects.filter(pipeline__id=id_pipeline, node_type=category):
        if node.node_type == 'Valores perdidos':...
        if node.node_type == 'Rellenar valores perdidos':...
        if node.node_type == 'Separar conjuntos':...
        if node.node_type == 'Naive Bayes':...
        if node.node_type == 'Random Forest':...
        if node.node_type == 'Support Vector Machine':...
        if node.node_type == 'Linear Regression':...
        if node.node_type == 'Precisión':...
return JsonResponse({"errors": errors}, status=200)
```

Figura 45. Fragmento vista run KoopaML

Así mismo la decisión de modificar un nodo sólo implica la modificación de ese nodo, gracias al encapsulamiento independiente de cada nodo estos cambios no afectarán al resto.

Lo mismo ocurre con la decisión de eliminar definitivamente un nodo.

De esta forma, mediante un sistema altamente cohesivo y con bajo nivel de acoplamiento que se ha descrito en los párrafos superiores, se pretende conseguir una aplicación flexible y escalable.

5.12. Despliegue

Para el despliegue de la aplicación en un servidor se ha hecho uso del servidor Gunicorn (gunicorn, s.f.), el servidor proxy Nginx (nginx, s.f.), un entorno virtual y una base de datos.

La configuración inicial de estos componentes es la siguiente:

- Nginx: para la configuración de Nginx se ha creado el fichero /etc/nginx/nginx.conf con el contenido que se muestra en la Figura 46.

```
server {
  server_name koopaml.grial.eu;
  charset   utf-8;
  # max upload size
  client_max_body_size 500M;  # adjust to taste
  location /.well-known {
          root /home/oper/KoopaML/code/koopaML;
  }
  # Django media
  location /media {
    alias /home/oper/KoopaML/code/koopaML/media;
  }
  location /static {
    alias /home/oper/KoopaML/code/koopaML/Desktop/static;
  }
  location /robots.txt {
    alias /home/oper/KoopaML/code/koopaML/Desktop/static/robots.txt;
  }
  location /favicon.ico {
    alias
/home/oper/KoopaML/code/koopaML/koopaML/static/img/faviconCARTIER.ico;
  }
  # Finally, send all non-media requests to the Django server.
  location / {
          include proxy_params;
   proxy_pass http://unix:/home/oper/KoopaML/koopa_ml.sock;
  }
  listen 443 ssl; # managed by Certbot
  ssl_certificate /etc/letsencrypt/live/koopaml.grial.eu/fullchain.pem; # managed by
Certbot
  ssl_certificate_key /etc/letsencrypt/live/koopaml.grial.eu/privkey.pem; # managed by
Certbot
  include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
  ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
}
server {
  if ($host = koopaml.grial.eu) {
    return 301 https://$host$request_uri;
  } # managed by Certbot
  server_name koopaml.grial.eu;
  listen 80;
  return 404; # managed by Certbot
```

Figura 46. Configuración Nginx

- Gunicorn: se ha de crear un dominio asociado a la IP del servidor, en este caso el dominio creado se ha denominado: koopaml.grial.eu.

  Se ha de crear también un fichero de configuración denominado /etc/systemd/system/gunicorn.service, cuyo contenido se puede observar en la Figura 47.

```
[Unit]
Description=koopaml gunicorn daemon
After=network.target
PartOf=gunicorn.target
ReloadPropagatedFrom=gunicorn.target
[Service]
User=oper
Group=www-data
WorkingDirectory=/home/oper/KoopaML/code/koopaML
ExecStart=/home/oper/KoopaML/venv/bin/gunicorn --timeout 1000 --access-logfile
/var/log/gunicorn/access-koopaml.log --error-logfile /var/log/gunicorn/error-
koopaml.log --log-level error --workers 4 --bind
unix:/home/oper/KoopaML/koopa_ml.sock koopaML.wsgi:application
[Install]
WantedBy=gunicorn.target
```

Figura 47. Configuración Gunicorn

- Entono virtual: inicialmente vacío.
- Base de datos: se crea una base de datos MongoDB con el nombre “koopaml” y una contraseña asociada.

Para desplegar la aplicación en el servidor creado se han seguido los siguientes pasos:

1. En el servidor se crea carpeta “koopaML” junto con un entorno virtual vacío, el cual se activa y se crea un archivo “requisitos.txt” con los requisitos para el correcto funcionamiento de la aplicación. Estos requisitos se obtienen de la salida del comando $ pip freeze. En el caso de la aplicación KoopaML los requisitos se muestran en la Figura 48.

```
asgiref==3.3.4
Django==3.0.5
django-cors-headers==3.7.0
djongo==1.3.4
joblib==1.0.1
jsonfield==3.1.0
nltk==3.2.4
numpy==1.20.3
pandas==1.2.5
preprocessing==0.1.13
pymongo==3.11.4
python-dateutil==2.8.1
pytz==2021.1
scikit-learn==0.24.2
scipy==1.7.0
six==1.16.0
sphinx-rtd-theme==0.2.4
sqlparse==0.2.4
threadpoolctl==2.1.0
```

Figura 48. Requisitos aplicación KoopaML

2. Tras ello se ejecuta el comando $ pip install -r requisitos.txt para instalarlos.
3. En este paso es necesario el código fuente de la aplicación, gracias a la utilización del sistema de controles Git explicado en el Capítulo 3. Técnicas y herramientas, es posible el acceso y la descarga de cualquier versión del código subida por cualquier miembro del equipo. Por tanto a través del enlace que referencia al código fuente a utilizar se utiliza el comando “git clone” y se descarga el código fuente.

   Es necesario realizar algunas modificaciones a este código para el correcto despliegue en el servidor, en concreto en el archivo settings.py se ha de indicar qué hosts tienen acceso al sistema, a través de la línea:

   ALLOWED_HOSTS = [‘IP_servidor’, ‘koopaml.grial.eu”]

   Es necesario también modificar el nombre de la base de datos para que coincida con el nombre de la nueva base de datos creada.
4. También debe crearse, a través del comando “createsuperuser” un nuevo super usuario para la aplicación.

Para ilustrar el procedimiento anteriormente descrito se ha realizado un diagrama de despliegue que puede consultarse en la Figura 49.

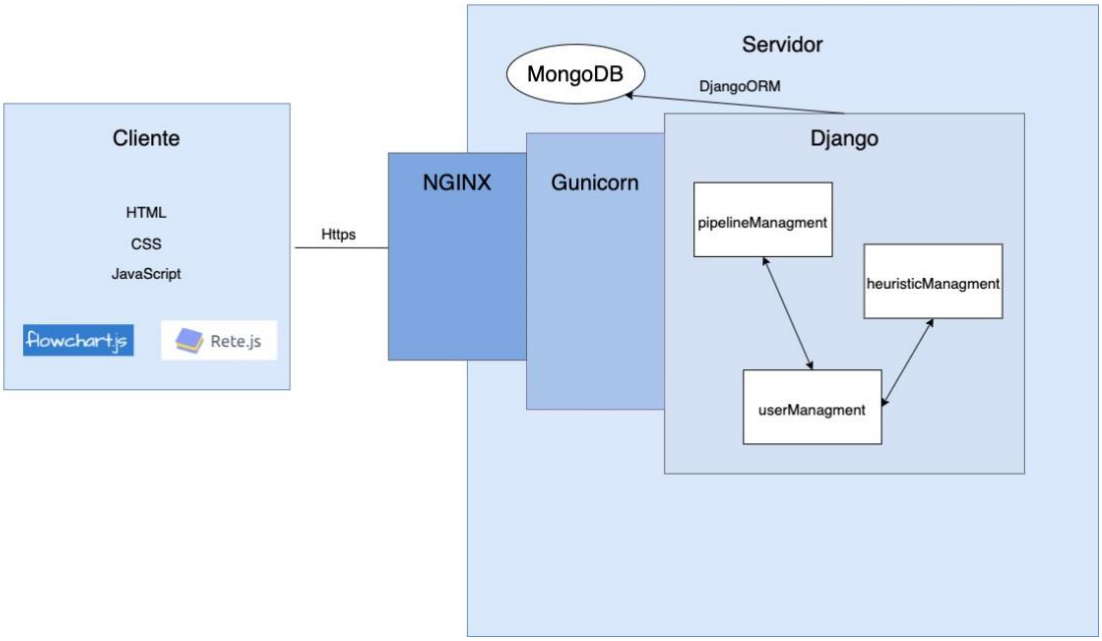


Figura 49. Diagrama de despliegue

Capítulo 6. Conclusiones

El aprendizaje automático, tal y como se explicó al comienzo del trabajo, es una disciplina que resulta de gran utilidad en muchos campos, entre ellos la medicina.

El principal problema afrontado durante este trabajo ha sido la concepción de una aplicación que acerque a los profesionales de la salud no expertos en programación y análisis de datos al mundo del aprendizaje automático, permitiéndoles utilizar sus conocimientos en medicina y la gran cantidad de datos que se generan en este campo para entrenar modelos de Machine Learning.

A pesar de existir múltiples aplicaciones (consultar apartado 5.2. Diseño centrado en el usuario) que tratan de acercar el aprendizaje automático a un público sin conocimientos técnicos, se decidió realizar una nueva aplicación adaptada al contexto concreto del departamento de cardiología del Hospital Universitario de Salamanca.

Esta decisión ha supuesto un proceso de desarrollo más largo, pero ha permitido construir una herramienta más personalizada, flexible y escalable. Esto ha sido posible gracias a la implementación de una infraestructura que fomenta el bajo acoplamiento y la alta cohesión a través del encapsulamiento de cada nodo del sistema, de forma que su modificación, eliminación o incorporación sea independiente del resto de nodos. Esta infraestructura facilita en gran medida la inclusión de otros nodos en el futuro, así como la modificación de los existentes según las necesidades de los usuarios.

Tal y como se ha descrito a lo largo del trabajo se ha incluido un enfoque de diseño centrado en el usuario, en el que, tras varias iteraciones, definiciones de la audiencia y creación de prototipos digitales, se ha obtenido como resultado una interfaz clara, sencilla y descrita por los usuarios de la audiencia objetivo como intuitiva. Además, la aplicación permite al usuario construir los flujos con total libertad, ofreciéndole en todo momento soporte en forma de indicaciones de errores, advertencias, visualizaciones disponibles, etc.

De esta forma se ha logrado crear una aplicación que ofrece soporte a usuarios sin conocimientos de aprendizaje automático sin que esto suponga limitar la libertad de creación de flujos de los usuarios.

Otros aspectos implementados derivados de los requisitos particulares de la aplicación, que han sido descritos con mayor detalle en el Capítulo 5. Aspectos relevantes, son la capacidad multilingüe de la aplicación, que aunque inicialmente ha sido pensada para los idiomas inglés y español es perfectamente extensible a cualquier otro idioma; la gestión de los archivos en el servidor; la actualización del flujo en tiempo real, lo que permite ofrecer el soporte en forma de iconos sobre errores, advertencias, etc.

Como resultado final de todo el proceso descrito a lo largo de este documento se ha obtenido una aplicación fácilmente escalable y flexible, multilingüe, con actualizaciones en tiempo real, con una eficiente gestión de archivos y de usuarios y con una interfaz clara que ofrece soporte a usuarios con pocos conocimientos de Machine Learning.

Por otro lado, desde el punto de vista personal este trabajo me ha permitido aplicar los conocimientos adquiridos en asignaturas como ingeniería del software, administración de sistemas e interacción persona-ordenador, además de haber aprendido un nuevo lenguaje de programación como es Python y haber ampliado mis conocimientos sobre lenguajes como HTML, CSS y JavaScript que únicamente había visto de forma parcial en la asignatura Interacción Persona-Ordenador. Así mismo he podido trabajar con diferentes frameworks y bibliotecas, enfrentándome así a la selección de herramientas y técnicas.

Otra experiencia nueva que me ha aportado el desarrollo de este trabajo ha sido la oportunidad de trabajar con clientes auténticos con necesidades reales.

Así mismo, al poner en práctica la metodología ágil Scrum, he podido observar paso a paso los resultados materiales de mi trabajo y aprendizaje, desde la concepción inicial de la idea hasta la culminación de ésta en forma de aplicación funcional desplegada en un servidor real.

6.1. Líneas futuras de trabajo

Durante todo este trabajo se ha resaltado la importancia de la infraestructura débilmente acoplada que soporta la edición de flujos de la aplicación, así como el encapsulamiento de las funcionalidades asociadas a cada nodo que permiten la modificación de los nodos existentes y la incorporación de nuevos nodos. No obstante, esta infraestructura que soporta la funcionalidad de edición de flujos de nodos no es la única estructura que permite la escalabilidad; la aplicación está diseñada para expandir

el alcance de otras funcionalidades, como la funcionalidad correspondiente a la gestión y uso de heurísticas, o la incorporación de tutoriales sobre aspectos básicos y avanzados de la aplicación.

6.1.1. Gestión de heurísticas

Actualmente la aplicación soporta la creación y edición de heurísticas en forma de pseudocódigo e ilustradas a través de un diagrama de flujo, como se expone en el apartado 5.5.2. Flowchart.js.

La creación y modificación de heurísticas está restringida a los usuarios cuyo rol es administrador o experto. Cabe destacar que no se permiten modificaciones en la heurística base, la cual ha sido desarrollada a partir del árbol de decisión de scikit learn (scikit learn, s.f.).

Estas heurísticas servirán en un futuro para ofrecer soporte a los usuarios en forma de sugerencias, utilizando el árbol de decisión definido en la heurística para aconsejar sobre qué algoritmo proporcionaría mejores resultados en función de los datos introducidos y los objetivos a conseguir.

6.1.2. Ejecución paso a paso

Actualmente la interfaz de la herramienta muestra un botón que permite la ejecución del flujo de forma continua, sin embargo, se planea incorporar la funcionalidad de ejecutar el flujo paso a paso. De esta forma la ejecución se producirá al ritmo que el usuario requiera, permitiéndole asimilar conceptos y comprobar los resultados de cada nodo antes de proseguir con la ejecución de los demás.

6.1.3. Plantilla flujo base

Debido al carácter escalable de la aplicación, se prevé la incorporación de numerosos nodos en el futuro, pudiendo resultar abrumador para un usuario que utilice por primera vez la aplicación. Tal y como arrojaron los resultados de la evaluación mediante grupo focal del prototipo (consultar apartado 5.2. Diseño centrado en el usuario para más información), los usuarios consideraron útil añadir un flujo básico o plantilla a la aplicación, de forma que pudiera servir como punto de inicio o ejemplo a los usuarios novatos.

6.1.4. Tutoriales

En la evaluación realizada a los usuarios a través de la técnica del grupo focal (consultar apartado 5.2. Diseño centrado en el usuario), los usuarios consideraron útil la incorporación de tutoriales o guías que cubrieran desde los aspectos más básicos de la aplicación hasta aspectos más avanzados. Por esta razón se prevé la realización de tutoriales en los que se muestren los pasos necesarios para alcanzar ciertos objetivos, que abarcarán distintos niveles de dificultad, desde los primeros pasos en la aplicación y la explicación de conceptos generales, hasta la construcción de flujos más complejos dependiendo de los nodos añadidos.

Esta funcionalidad se expandirá en paralelo a la incorporación de nuevos nodos, con el objetivo de ofrecer soporte a los usuarios de forma visual.

Bibliografía

Abellán, E. (2020, Marzo 5). *Scrum: qué es y cómo funciona esta metodología*. Retrieved from We Are Marketing: https://www.wearemarketing.com/es/blog/metodologia-scrum-que-es-y-como-funciona.html#:~:text=La%20metodolog%C3%ADa%20Scrum%20es%20un,equipos%20que%20manejan%20proyectos%20complejos.&text=Esto%20permite%20al%20cliente%2C%20junto,obtener%20ventas%20(Sales%2

Adobe. (2021). *What is Adobe XD and what is it used for?* Retrieved from Adobe products: https://www.adobe.com/products/xd/learn/get-started/what-is-adobe-xd-used-for.html

Amezcua, M. (2016). 12 Golden Rules for conducting a Focus Group. *Index de Enfermería*.

Atlassian. (n.d.). *La herramienta de desarrollo de software líder de los equipos ágiles*. Retrieved from Atlassian: https://www.atlassian.com/es/software/jira

Bacinger, T. (n.d.). *What is Bootstrap? A Short Bootstrap Tutorial on the What, Why, and How*. Retrieved from Developers: https://www.toptal.com/front-end/what-is-bootstrap-a-short-tutorial-on-the-what-why-and-how

Bowman, D. (2014). *The psychology of color in web design*. Retrieved from 99designs: https://99designs.es/blog/creative-inspiration/psychology-color-web-design/

Cantú, A. (2018). *Qué es: Carga Cognitiva*. Retrieved from Intuitivamente.: https://blog.acantu.com/que-es-carga-cognitiva/

Chromatic circle or color wheel. (2019, Enero 15). Retrieved from THE CHROMATIC CIRCLE : https://colorofmeaning.blogspot.com/2019/01/the-chromatic-circle.html

Costa, R. (2020, Marzo 5). *How to design user scenarios: best practices and examples*. Retrieved from justinmind: https://www.justinmind.com/blog/how-to-design-user-scenarios/

Docs, MDN Web. (2021). *Django introduction*. Retrieved from developer mozilla: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction

Documentation, django. (n.d.). *Managing static files (e.g. images, JavaScript, CSS)*. Retrieved from django: https://docs.djangoproject.com/en/3.2/howto/static-files/

Flannery, M. (2019, Agosto 15). *5 Reasons to Use Adobe XD for Your Next Project*. Retrieved from imarc: https://www.imarc.com/blog/5-reasons-to-use-adobe-xd

Garcia Peñalvo, F., & Garcia Holgado, A. (2017). FUNDAMENTOS DE LA VISTA DE CASOS DE USO. Salamanca, Castilla y León, España.

Google. (n.d.). *Google Colab*. Retrieved from https://colab.research.google.com/

Google. (n.d.). *Material Design*. Retrieved from https://material.io/design

GRAPH. (n.d.). *Machine Learning en Python*. Retrieved from GRAPH: https://www.grapheverywhere.com/machine-learning-en-python/

gunicorn. (n.d.). *gunicorn*. Retrieved from gunicorn: https://gunicorn.org/

Hamui-Sutton, A., & Varela-Ruiz, M. (2013). La técnica de grupos focales. *Investigación en Educación Médica*. Retrieved from https://doi.org/10.1016/S2007-5057(13)72683-8

Hannah, J. (2021, Junio 23). *What Exactly Is Wireframing? A Comprehensive Guide*. Retrieved from CAREERFOUNDRY: https://careerfoundry.com/en/blog/ux-design/what-is-a-wireframe-guide/

IBM Cloud Education. (2020, Julio 15). *Aprendizaje automático*. Retrieved from IBM: https://www.ibm.com/cloud/learn/machine-learning

Interaction Design Foundation. (n.d.). *User Scenarios*. Retrieved from Interaction Design Foundation: https://www.interaction-design.org/literature/topics/user-scenarios

Keras. (n.d.). *Keras*. Retrieved from Keras: https://keras.io/

KNIME. (n.d.). *KNIME Analytics Platform*. Retrieved from https://www.knime.com/knime-analytics-platform

LearningML. (2021). *LearningML - AI made easy*. Retrieved from https://web.learningml.org/

Machine Learning for Kids. (n.d.). *Enséñale a una computadora a jugar un juego*. Retrieved from https://machinelearningforkids.co.uk/

MAHOUT. (2021). *MAHOUT*. Retrieved from MAHOUT: https://mahout.apache.org/

Martín Abadi, P. B. (2016). TensorFlow: A System for Large-Scale Machine Learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16).* Savannah, GA, USA: usenix. Retrieved from https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf

Materials, N. (2019). Ascent of machine learning in medicine. *Nature Materials, 18*(407).

MongoDB Atlas. (n.d.). *MongoDB Atlas*. Retrieved from MongoDB Atlas: https://www.mongodb.com/cloud/atlas/lp/try2?utm_source=google&utm_campaign=gs_emea_spain_search_core_brand_atlas_desktop&utm_term=mongo%20db&utm_medium=cpc_paid_search&utm_ad=e&utm_ad_campaign_id=12212624563&gclid=CjwKCAjwoNuGBhA8EiwAFxomAyJbn_ERAbRVjS5Xe

mro. (2020, Julio 4). *Modelo de Arquitectura C4*. Retrieved from Wordpress: https://mauro.ec/2020/07/04/modelo-de-arquitectura-c4/

nginx. (n.d.). *nginx*. Retrieved from nginx: https://www.nginx.com/

Nielsen, J. (2020, Noviembre 15). *10 Usability Heuristics for User Interface Design*. Retrieved from Nielsen Norman Group: https://www.nngroup.com/articles/ten-usability-heuristics/

Nielsen, L. (2010). Chapter 30. Personas. In L. Nielsen, *The Encyclopedia of Human-Computer Interaction, 2nd Ed.* Interaction Design Foundation.

Pesheva, E. (2019, Abril 3). *El doctor y la maquina*. Retrieved from Harvar Medical School: https://hms.harvard.edu/news/doctor-machine

PyTorch. (n.d.). *PyTorch*. Retrieved from PyTorch: https://pytorch.org/

Qiany, S. (2020, Septiembre 23). *A Personas Guideline, From What They Are to How To Use*. Retrieved from UX Collective: https://uxdesign.cc/while-we-are-talking-about-personas-what-exactly-are-we-talking-525a645eb61a

QUICKSPROUT. (2019, Junio 10). *5 Easy Steps to Creating a Sitemap For a Website*. Retrieved from QUICKSPROUT: https://www.quicksprout.com/creating-website-sitemap/

rapidminer. (2021). *Get started with RapidMiner*. Retrieved from rapidminer: https://rapidminer.com/get-started/

Rete.js. (n.d.). *Rete.js*. Retrieved from The JavaScript framework for visual programming: https://rete.js.org/#/

Rete.js. (n.d.). *Rete.js example basic*. Retrieved from Rete.js : https://rete.js.org/#/examples/basic

Rikke Friis, D., & Yu Siang, T. (2021, Enero). *Personas – A Simple Introduction*. Retrieved from Interaction Design Foundation: https://www.interaction-design.org/literature/article/personas-why-and-how-you-should-use-them

Sanyal, S. (2019, Mayo 1). *5 Reasons Why Doctors Should Learn Data Science*. (Forbes) Retrieved from Forbes: https://www.forbes.com/sites/shourjyasanyal/2019/05/01/5-reasons-why-doctors-should-learn-data-science/?sh=62b3344b2b85

scikit learn. (n.d.). *Choosing the right estimator*. Retrieved from scikit learn: https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

scikit-learn. (n.d.). *scikit-learn*. Retrieved from scikit learn: https://scikit-learn.org/stable/

Sordo, A. I. (2021, Junio 29). *Las 15 tipografías para web más recomendadas (y que deberías descargar)*. Retrieved from Hubspot: https://blog.hubspot.es/marketing/tipografias-para-web

Therón, R. (2020). Interacción Persona Ordenador. Salamanca, Castilla y León, España.

Tsemashko, I., I. Kolokolchikova, & O. Titova. (2017). *The Psychology of Colour in Marketing and Branding.* Ucrania: Tavria State Agrotechnological University.

Urquiaga, J. C. (n.d.). *¿Por qué aprender Python?* Retrieved from DevCode: https://devcode.la/blog/por-que-aprender-python/

VIEWNEXT. (2019). SCRUM FRAMEWORK. Salamanca, Castilla y León, España.

Vila, X. G. (2012, Mayo 22). *Consejos para redactar cuestionario*. Retrieved from ime: https://www.investigacionmercados.es/consejos-para-redactar-cuestionarios/

Weka 3: Machine Learning Software in Java . (n.d.). *Weka 3: Machine Learning Software in Java*. Retrieved from Weka 3: Machine Learning Software in Java : https://www.cs.waikato.ac.nz/ml/weka/

Wikipedia. (2020, Julio 29). *GitLab*. Retrieved from Wikipedia: https://es.wikipedia.org/wiki/GitLab

yeeply. (n.d.). *Diseño de apps: La importancia de la tipografía*. Retrieved from yeeply: https://www.yeeply.com/blog/diseno-de-apps-la-importancia-de-la-tipografia/

Zoom. (2021). *Zoom*. Retrieved from Zoom: https://zoom.us/