ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

Máster en Big Data: Tecnología y Analítica Avanzada

Desarrollo de una librería para el cálculo de modelos de clasificación y su evaluación en paralelo en la nube

Autor
Antonio González Suárez

Dirigido por
Guillermo Valle Gutiérrez

Madrid
26 de junio de 2023

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título: Desarrollo de una librería para el cálculo de modelos de clasificación y su evaluación en paralelo en la nube

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico 2022-2023 es de mi autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Fdo.: Antonio González Suárez Fecha: 26/06/2023

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Guillermo Valle Gutiérrez Fecha: 26/06/2023

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESINAS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D.Antonio González Suárez **DECLARA** ser el titular de los derechos de propiedad intelectual de la obra: Desarrollo de una librería para el cálculo de modelos de clasificación y su evaluación en paralelo en la nube, que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, los derechos de digitalización, de archivo, de reproducción, de distribución y de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

(a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar "marcas de agua" o cualquier otro sistema de seguridad o de protección.

(b) Reproducirla en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.

(c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.

(d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.

(e) Asignar por defecto a estos trabajos una licencia Creative Commons.

(f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente*).

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

(a) Que la Universidad identifique claramente su nombre como autor de la misma

(b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.

(c) Solicitar la retirada de la obra del repositorio por causa justificada.

(d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos de propiedad intelectual sobre ella.

5º. Deberes del autor.

(a) El autor se compromete a:

(b) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.

(c) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.

(d) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que pudieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e intereses a causa de la cesión.

(e) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción de derechos derivada de las obras objeto de la cesión.

6º. Fines y funcionamiento del Repositorio Institucional.

La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.
- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusiva del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.
- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.
- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados o en caso de reclamaciones de terceros.

Madrid, a 26 de Junio de 2023

ACEPTA

Fdo.: Antonio González

Motivos para solicitar el acceso restringido, cerrado o embargado del trabajo en el Repositorio Institucional:

Resumen

El presente trabajo propone una solución para abordar las necesidades actuales de las empresas relacionadas con la implementación del aprendizaje automático en sus flujos de trabajo.

Con este objetivo en mente, se ha desarrollado AutoML (Automated Machine Learning), un módulo de Python que automatiza el proceso de creación de modelos de *machine learning*. Esta herramienta permite a las empresas ahorrar tiempo y recursos al eliminar la necesidad de realizar el desarrollo manual de modelos.

Además, para hacer que AutoML sea fácil de usar y accesible para los usuarios, se ha diseñado y construido una infraestructura completa (ver Figura 1) que permite interactuar con el módulo a través de una aplicación web. Esto significa que no se requiere la instalación de software adicional, lo que simplifica aún más su implementación en los flujos de trabajo existentes.

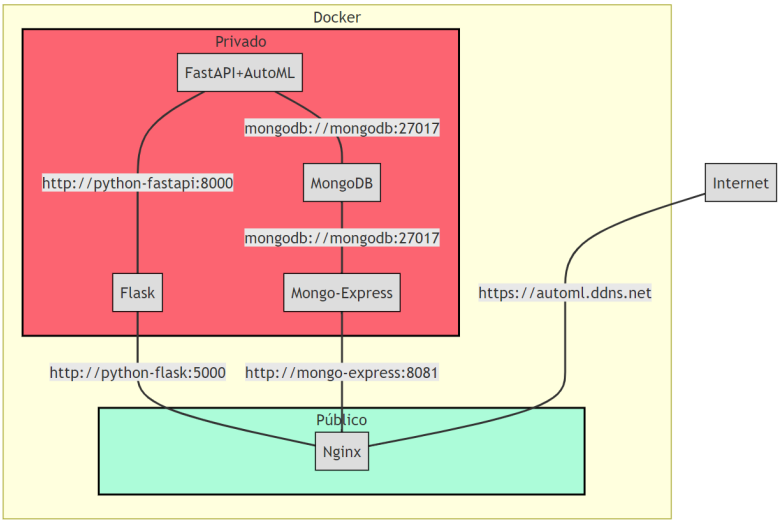

Figura 1: Arquitectura de la solución técnica

Abstract

This paper proposes a solution to address the current needs of companies related to the implementation of machine learning in their workflows.

With this goal in mind, we have developed AutoML (Automated Machine Learning), a Python module that automates the process of creating machine learning models. This tool allows companies to save time and resources by eliminating the need for manual model development.

Furthermore, to make AutoML user-friendly and accessible, we have designed and built a comprehensive infrastructure (see Figure 2) that enables interaction with the module through a web application. This means that no additional software installation is required, further simplifying its implementation in existing workflows.

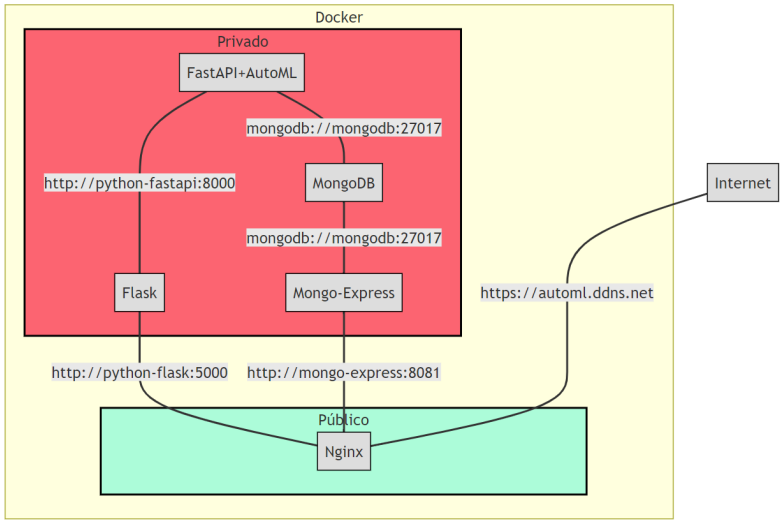

Figura 2: Architecture of the technical solution

Índice general

Índice de figuras

Acrónimos y siglas

API	Interfaz de programación de aplicaciones
AutoML	Automated Machine Learning
GCP	Google Cloud Platform
HTTP	Protocolo de transferencia de hipertexto
HTTPS	Protocolo de transferencia de hipertexto seguro
ICAI	Instituto Católico de Artes e Industrias
IP	Protocolo de internet
ML	Aprendizaje automático
SaaS	Software como Servicio

Capítulo 1

Introducción

El aprendizaje automático automatizado, también conocido como AutoML (*Automated Machine Learning*) ha ganado atención significativa en los últimos años como un enfoque prometedor para democratizar el aprendizaje automático y hacerlo más accesible a usuarios con conocimientos limitados en ciencia de datos. Su objetivo es poder simplificar y agilizar la construcción de modelos de aprendizaje automático, de tal forma que el científico de datos pueda centrarse en el diseño del problema y la interpretación de resultados, mientras que gran parte del trabajo técnico es automatizado y realizado por la herramienta.

1.1. Descripción del problema

El proyecto presenta una solución al problema de automatizar el entrenamiento y despliegue de modelos de aprendizaje automático. En general, el problema principal es el nivel de conocimiento técnico necesario para poder implementar modelos de aprendizaje automático es relativamente alto debido a la necesidad de conocer lenguajes de programación como Python o similares.

El entrenamiento de modelos de aprendizaje automático requiere del uso de lenguajes como Python y en particular el conocimiento profundo de librerías como *scikit-learn* y algoritmos de aprendizaje. Esto supone una gran barrera a la hora de implementar aprendizaje automático en los flujos de trabajo de un negocio.

Además, en la mayoría de casos las tareas que se realizan de preprocesado y modelado suelen ser repetitivas y por tanto su automatización dejaría más tiempo para el análisis de modelos y resultados.

Por ello, se propone una solución llamada AutoML a través de una aplicación web en la url `https://automl.ddns.net`. Mediante una interfaz cómoda y sencilla

se facilita la tarea de gestión de tanto conjuntos de datos, como de modelos de aprendizaje automático.

Actualmente, la herramienta permite las siguientes tareas:

- **Gestión de usuarios:** Permite generar de manera automática nuevos usuarios en la plataforma y aísla la información entre ellos, es decir, cada usuario solo tiene permisos para ver sus datos.
- **Análisis del conjunto de datos:** Se ofrece un reporte inicial básico de los datos y un análisis exploratorio profundo del conjunto de datos subido a la plataforma.
- **Entrenamiento de modelos:** Generación automática de modelos optimizados (OrangePrints) para el problema a resolver.
- **Predicción:** Realizar desde la aplicación predicciones para nuevos datos.

En esta primera versión de la herramienta, solamente se permite resolver problemas de clasificación. Sin embargo, en un futuro se planea añadir funcionalidades para resolver tanto problemas de regresión como de series temporales.

1.2. Motivación

Cada día es más frecuente ver cómo modelos de aprendizaje automático se integran de manera profunda en el tejido empresarial[7]. Desde las áreas técnicas hasta las estratégicas, estos modelos se han convertido en herramientas fundamentales para implementar soluciones empresariales y pronosticar resultados futuros, que mejoran la toma de decisiones.

No obstante, aprovechar al máximo estos modelos puede resultar complicado para aquellos con sólidos conocimientos en el ámbito empresarial pero escasa experiencia técnica. Por este motivo, se hace imprescindible la democratización de las herramientas de aprendizaje automático.

La democratización de estas herramientas de *machine learning* implica hacerlas accesibles y amigables, de manera que cualquier persona con conocimientos empresariales pueda aprovechar su potencial. Esto no solo agiliza la adopción de estas tecnologías en el entorno empresarial, sino que también fomenta la creatividad y la colaboración, permitiendo a diferentes departamentos y equipos experimentar y desarrollar soluciones personalizadas.

1.3. Objetivos del trabajo

Los objetivos que se han planteado para esta versión de la aplicación web son:

- Automatización del análisis exploratorio de los datos.
- Automatización de la creación de modelos de aprendizaje automático.
- Despliegue y puesta en producción de dichos modelos.
- Facilitar la gobernanza de modelos de *Machine Learning*.

1.4. Estructura del documento

La memoria se encuentran dividida en los siguientes capítulos:

- **Estado del arte**: Introducción a las soluciones actuales para la democratización de los modelos de aprendizaje automático.
- **Desarrollo del backend en Python**: Documentación de la librería desarrollada en Python que actúa como motor de la aplicación web.
- **Despliegue en la nube**: Descripción de la arquitectura utilizada durante el despliegue en la nube e implementación técnica mediante el marco de trabajo de *Flask* y *FastAPI*
- **Conclusiones y trabajos futuros**: Capítulo con las reflexiones sobre el proyecto y discusión sobre los posibles evolutivos a introducir en el futuro.

Capítulo 2

Estado del arte

En este capítulo, se detallarán las distintas tecnologías existentes en la actualidad para la democratización del aprendizaje automático (*Machine Learning*), tanto aquellas opciones de pago como las gratuitas[6].

Representando las herramientas de pago, destacan las siguientes:

- **H20 Driverless AI**: Esta herramienta de pago ofrece una amplia gama de funcionalidades y soporte técnico. Operando bajo el modelo de Software as a Service (SaaS), proporciona una experiencia sencilla a través de una interfaz visual intuitiva. Se explorarán las características principales de H20 Driverless AI y se analizarán los costos de licenciamiento asociados.
- **DataRobot**: Otra herramienta de pago que brinda soluciones en el campo del aprendizaje automático. Al igual que H20 Driverless AI, DataRobot se enfoca en facilitar la creación y el despliegue de modelos de aprendizaje automático sin requerir un conocimiento profundo de programación. Se examinarán las características principales de DataRobot y se evaluarán los precios de licenciamiento correspondientes.

En cuanto a las herramientas gratuitas, se explorará específicamente la librería de Python **Pycaret**[10]. Se analizarán las funcionalidades que ofrece, su flexibilidad y su capacidad para facilitar la implementación de modelos de aprendizaje automático sin costo adicional.

2.1. Software con licencia

En esta sección, se discutirá sobre las herramientas que cuentan con licenciamiento y soporte técnico. Estas plataformas, en general, siguen el modelo de SaaS y se

distinguen por ofrecer una experiencia simplificada a través de interfaces visuales claras y amigables.

Se explorarán las características principales de cada una de estas herramientas de pago, así como los costos asociados a su licenciamiento.

2.1.1. H20 Driverless AI

H20 Driverless AI es una herramienta de AutoML que se destaca por su enfoque en la automatización avanzada y la simplificación del proceso de construcción de modelos de aprendizaje automático. Con su interfaz intuitiva, permite a los usuarios, incluso aquellos sin experiencia previa en aprendizaje automático, desarrollar y desplegar modelos de manera eficiente.

La automatización avanzada de H20 Driverless AI abarca desde la selección de variables hasta la optimización de hiperparámetros. La herramienta realiza la ingeniería de variables automáticamente, utilizando técnicas como el procesamiento de lenguaje natural (NLP) y el análisis de series temporales para extraer características relevantes de los datos.

H20 Driverless AI también destaca por su enfoque en la explicabilidad y la interpretación de modelos. Proporciona explicaciones detalladas sobre cómo se toman las decisiones en los modelos generados, lo que resulta especialmente útil en aplicaciones donde la explicabilidad es necesaria para cumplir con regulaciones o políticas internas.

Coste de licencias

Según IBM[12] a fecha de 2019 y, válidos en Estados Unidos, los precios de licencias de esta solución son los siguientes:

- Driverless AI Single user 3 year subscription: 390.000$
- Driverless AI Single user 5 year subscription: 650.000$
- Driverless AI Min Qty 3 users 3 year subscription: 300.000$
- Driverless AI Min Qty 3 users 5 year subscription: 500.000$
- Driverless AI Single user 1 year subscription with GPU: 170.000$
- Driverless AI Single user 3 years subscription with GPU: 510.000$
- Driverless AI Single user 5 years subscription with GPU: 850.000$

2.1.2. DataRobot

DataRobot es una herramienta de AutoML centrada en la democratización del ML, es decir, su automatización y simplificar el desarrollo de modelos de aprendizaje automático. Con su interfaz visual y amigable, permite a los usuarios explorar, preparar y modelar los datos de manera intuitiva, sin necesidad de tener un conocimiento profundo de algoritmos o programación.

La automatización del proceso de aprendizaje automático es una fortaleza destacada de DataRobot. La herramienta automatiza diversas tareas complejas, como la selección de algoritmos, la ingeniería de características y la optimización de hiperparámetros. Esto acelera el tiempo de desarrollo y permite a los usuarios obtener rápidamente modelos de alta calidad.

DataRobot también se destaca por su capacidad para gestionar múltiples algoritmos y técnicas de aprendizaje automático. La herramienta ofrece una amplia gama de opciones de modelado y permite a los usuarios comparar y evaluar diferentes algoritmos para encontrar el más adecuado para su problema específico.

Uno de sus puntos fuertes con respecto a los competidores es la sección de despliegues. Esta permite una monitorización de los modelos y facilidad para realizar predicciones a través de una API. Algunas de las características de esta sección son:

- **Data Drift:** Basándose en la distribución de los datos con los que el modelo fue entrenado, es capaz de determinar si la distribución de alguna variable está cambiando de forma sustancial en las predicciones. De esta manera, lanza una alarma avisando que las predicciones pueden dejar de ser fiables ya que el modelo nunca ha observado esos valores.
- **Metrics over time:** Durante las predicciones a cada muestra se le asigna un valor único (PK), si en un futuro se tiene acceso al valor real que ha tenido dicha predicción se puede subir a la plataforma y se calculan las métricas en las predicciones. Con esto se puede llegar a observar posibles bajadas de rendimiento en las predicciones con respecto a las métricas de entrenamiento.

Coste de licencias

Para los precios de DataRobot solo se conocen las licencias tipo SaaS disponibles en las plataformas de computación en la nube Amazon Web Services y Google Cloud Platform:

- DataRobot AI Cloud for AWS (Starter Pack for AutoML, AutoTS, MLOps; 5 users; standard support; 1 year): 98000$ [2]

- DataRobot AI Cloud Platform for Google Cloud: 8166.67$ mensuales [8]

2.2. Software libre

En el contexto de las herramientas gratuitas para la democratización del aprendizaje automático, existen diversas librerías de Python que ofrecen funcionalidades de AutoML. Una de las opciones más destacadas es Pycaret.

2.2.1. Pycaret

Pycaret [1] es una librería de Python de código abierto que ofrece funcionalidades de AutoML para simplificar y acelerar el desarrollo de modelos de aprendizaje automático. A diferencia de las herramientas de pago, Pycaret es una opción gratuita y de fácil acceso para los desarrolladores y científicos de datos.

Una de las principales ventajas de Pycaret es su capacidad para automatizar muchas tareas repetitivas y complejas en el proceso de construcción de modelos. La librería proporciona una amplia gama de algoritmos y técnicas de preprocesamiento de datos, lo que permite a los usuarios entrenar y comparar varios modelos con solo unas pocas líneas de código. Pycaret se encarga automáticamente de tareas como la selección de características, el ajuste de hiperparámetros y la evaluación del rendimiento del modelo.

Pycaret también facilita la implementación y el despliegue de modelos entrenados. Proporciona funcionalidades para exportar modelos en formato *pickle* y permite su reutilización en diferentes entornos.

Sin embargo, Pycaret tiene algunas limitaciones en comparación con las herramientas de pago. Por ejemplo, puede tener un conjunto más limitado de algoritmos y funcionalidades avanzadas en comparación con las soluciones comerciales, además necesita de conocimientos de Python ya que no proporciona una interfaz web.

Capítulo 3

Desarrollo del backend en Python

En este capítulo se describe la implementación técnica del *backend* de la aplicación web.

AutoML se ha desarrollado como una librería en Python, con el objetivo de permitir a los usuarios con pocos conocimientos en programación crear modelos de manera sencilla. El diseño de la librería se divide en dos secciones: una pública y otra privada.

La sección pública es la interfaz con la que los usuarios interactúan directamente. Se han creado dos clases principales que el usuario utilizará:

- Dataset
- AutoML

Estas clases utilizan funcionalidades diseñadas en la sección privada del código.

La separación en estas secciones tiene como propósito reducir la cantidad de código con la que el usuario interacciona directamente.

3.1. Sección pública

3.1.1. Dataset

Dataset es una clase de Python que permite enriquecer el concepto de *DataFrame* de la librería *Pandas*. Los principales atributos que tendremos accesibles en una instancia de Dataset son:

- name: Nombre del dataset para poder ser referenciado en múltiples experimentos.

- data: El conjunto de datos en sí.
- target: La columna que contiene la información del objetivo.
- target_name: El nombre de la columna con el objetivo.
- problem_type: Tipo de problema que se desea resolver. (En esta versión solamente clasificación)

Para facilitar la inicialización de la clase Dataset a los usuarios se han implementado los siguientes métodos para construir la instancia:

- from_dataframe: Permite crear un Dataset directamente desde un *Dataframe*. Como argumentos adicionales toma el nombre de la variable objetivo, qué tipo de problema se trata y el nombre que se le desea dar al Dataset.
- from_csv: Similar a from_dataframe, pero en este caso permite cargar los datos desde un CSV indicando cual es el carácter de separación de campos.

Una vez creada la instancia, el usuario deberá utilizarla para crear un experimento con la clase AutoML.

3.1.2. AutoML

AutoML es la clase principal de la librería, en ella se configura el experimento que se desea realizar. Es la encargada de generar, entrenar y analizar tantos modelos como el usuario desee configurar.

Para inicializar una instancia de AutoML son necesarios los siguientes elementos:

- Dataset: Una instancia de la clase Dataset que se utilizará para realizar los entrenamientos.
- config: Es un JSON con la configuración que el usuario desee para el experimento. Los campos que admite actualmente AutoML son:
  - name: Nombre que se le desee dar al experimento.
  - preprocessing: Permite la configuración de los preprocesados, debe ser un JSON con la siguiente estructura:
    - numeric_preprocessors: Lista con los preprocesados numéricos.
    - categorical_preprocessors: Lista con los preprocesados categóricos.
    - numeric_imputers: Lista con los métodos de imputación que se desean probar.
    - categorical_imputers: Lista con los métodos de imputación categóricos.

El único campo estrictamente necesario es el de *name*, si el usuario no introduce ninguna configuración extra, se configura de forma automática mediante la clase OrangePrint (3.3.1)

Una vez inicializada esta clase tiene tres métodos que pueden ser utilizados por el usuario: **fit**, **show_results** y **predict**.

El método **fit** hace uso de los distintos modelos que ha construido OrangePrint para entrenar y generar métricas de cada uno de ellos. Una vez entrenados se pueden mostrar los resultados mediante **show_results**. Al mostrarse, cada uno de los modelos tiene un nombre para identificarlo.

Por último, se pueden realizar predicciones para nuevos datos con **predict**. Como datos de entrada es necesario un *DataFrame* con una estructura similar a la utilizada durante el entrenamiento y el identificador del modelo obtenido al visualizar los resultados.

3.2. Flujo de uso en Python

A continuación se muestra una demo de código para hacer uso del software desarrollado a través de Python:

```
import pandas as pd
from automl import AutoML, Dataset

# Creacion instancia de Dataset
ds = Dataset.from_csv(
            path="test.csv",
            target="y",
            problem_type="classification",
            )

# Experimento con configuracion automatica de OrangePrint
auto_config = {
    "name": "Experimento de prueba sin configuracion"
}
experimento_automatico = AutoML(
            dataset=ds,
            config=auto_config
            )


# Ejemplo de experimento personalizado
config_personalizada = {
    "name": "Experimento de prueba personalizado",
    "preprocessing":{
```

```
        "numeric_preprocessors": ["MinMaxScaler","passthrough"],
        "categorical_preprocessors": ["OneHotEncoder","
    OrdinalEncoder"],
        "numeric_imputers": ["SimpleImputer"]
    }
}
experimento_personalizado = AutoML(
            dataset=ds,
            config=config_personalizada
            )

# Usando el experimento por defecto (experimento_automatico)
# Entrenamiento de modelos
experimento_automatico.fit()

# Mostrar resultados
experimento_automatico.show_results()

# Prediccion de nuevo dataset
new_data = pd.read_csv("new_data.csv")
prediccion = experimento_automatico.predict(
    pipeline_id="MinMaxScaler_OneHotEncoder_DecisionTreeClassifier
    ", # Se obtiene de la tabla de experimento_automatico.
    show_results()
    df_to_predict=new_data
    )
```

3.3. Sección privada

En la sección privada, se encuentra la clase OrangePrint, que es la funcionalidad más relevante y no está destinada a interaccionar directamente con el usuario. Además, existen otros elementos que facilitan la creación de la librería pero no son relevantes para esta documentación.

3.3.1. OrangePrint

La clase OrangePrint juega un papel crucial en la implementación de AutoML. Su función principal es generar los distintos modelos que serán entrenados posteriormente.

Un modelo se puede dividir en dos bloques principales:

- Preprocesamiento
- Modelo

El preprocesamiento se puede separar en variables numéricas y variables categóricas. Además, para cada tipo de variable se pueden realizar diferentes pasos de preprocesamiento. Por ejemplo, para las variables numéricas es común realizar un escalado y la imputación de valores nulos.

El amplio abanico de posibilidades puede hacer que la creación de modelos sea compleja para aquellos sin conocimientos en Machine Learning. OrangePrint se encarga de construir las mejores combinaciones de preprocesamiento y modelos para entregárselas a la clase AutoML, donde serán entrenadas posteriormente.

Además, OrangePrint realiza una optimización inicial para determinar qué combinaciones tienen más probabilidad de éxito, dependiendo del tipo de problema.

Capítulo 4

Despliegue en la nube

La librería AutoML se puede utilizar haciendo uso de un interprete de Python o de cuadernos de Jupyter siempre que las dependencias se encuentren instaladas. Sin embargo, esto puede ser un inconveniente para los usuarios menos habituados al desarrollo de modelos de Machine Learning.

Es por ello que se ha planteado el diseño de una aplicación web que interaccione con AutoML.

En este capítulo, se presentarán las tecnologías empleadas para el despliegue del servicio en la nube, así como posibles mejoras a considerar. También se abordarán temas relevantes relacionados con la ciberseguridad del sistema.

La descripción del proceso de despliegue se puede dividir en dos grandes bloques:

- **Arquitectura del sistema:** En esta sección se abordará la estructura general del sistema y las tecnologías empleadas para su funcionamiento en la nube.
- **Aplicación web:** Esta parte se centra en la interacción del usuario con la librería AutoML sin requerir conocimientos de Python.

4.1. Arquitectura del sistema

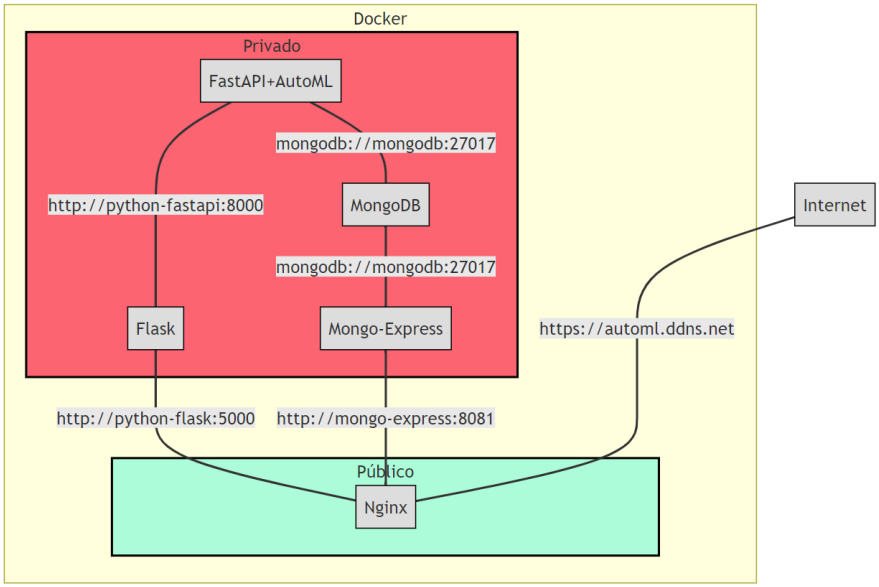

Figura 4.1: Diagrama con la arquitectura final de contenedores desplegada con *Dockers*

El despliegue de la aplicación se realizará haciendo uso de la tecnología de *Dockers*. Para garantizar la seguridad de la información almacenada en la base de datos se ha optado por un diseño con múltiples contenedores. A grandes rasgos, se pueden separar entre un grupo privado y otro público. El grupo privado no tiene conexión directa con Internet, es decir, es inexistente para cualquier persona no administradora del sistema y, el grupo público consiste únicamente del *docker* de *Nginx* que actúa como proxy inverso.

En está sección se tratará solamente sobre el grupo público, en el resto de contenedores se profundizará en el apartado 4.2.

Nginx

Nginx es un servidor web de alto rendimiento que también funciona como un proxy[1] inverso. Un proxy inverso actúa como intermediario entre los clientes y los servidores, recibiendo las solicitudes de los clientes y reenviándolas a los servidores correspondientes. A diferencia de un proxy tradicional, que protege la identidad

[1]Servidor proxy: Servidor que hace de intermediario entre la peticiones de un cliente y el servidor final. Habitualmente se encuentra del lado del cliente

de los clientes, un proxy inverso protege la identidad de los servidores al ocultar su ubicación y exponer solo su propia dirección.

Esto mejora la seguridad, el rendimiento y la escalabilidad del sistema, ya que el proxy inverso puede realizar funciones como el balanceo de carga y el caché para optimizar el flujo de tráfico y garantizar un acceso rápido a los recursos del servidor.

El motivo principal de ser utilizado en este proyecto es el de ocultar al exterior las ubicaciones de los servicios del grupo de contenedores privado. De esta forma, se controla perfectamente que usuarios (IPs) acceden a que ubicaciones, posibilitando el bloqueo de servicios en función de la IP actuando como un *firewall*.

En este caso Nginx solo tiene dos conexiones directas desde el grupo privado: *Flask* a través de la url http://python-flask:5000 y *Mongo-Express* en http://mongo-express:8081. Estás direcciones son solamente accesibles desde la red interna de *dockers*, bloqueando las interacciones directas desde el exterior con estos servicios.

Para dar visibilidad a dichos servicios Nginx los disponibiliza en diferentes ubicaciones y securiza las conexiones con el exterior mediante certificados generados explícitamente para `https://automl.ddns.net`

Certificados HTTPS

Los certificados HTTPS son archivos digitales que se utilizan para establecer una conexión segura entre un servidor web y un navegador. Proporcionan autenticación y cifrado de datos, lo que garantiza que la comunicación entre el usuario y el sitio web sea segura y confidencial.

Un certificado HTTPS incluye información sobre el propietario del sitio web (como el nombre de dominio) y la entidad que lo emite, también conocida como Autoridad de Certificación (CA). La CA es una organización confiable que verifica la identidad del propietario del sitio web y emite el certificado. Algunas CAs conocidas son Symantec, Comodo, Let's Encrypt, entre otras.

Let's Encrypt es una CA gratuita y automatizada que se ha vuelto muy popular debido a su enfoque en facilitar el proceso de obtención y renovación de certificados HTTPS. A diferencia de otras CAs, Let's Encrypt permite solicitar y obtener certificados de forma automatizada, lo que simplifica significativamente el proceso.

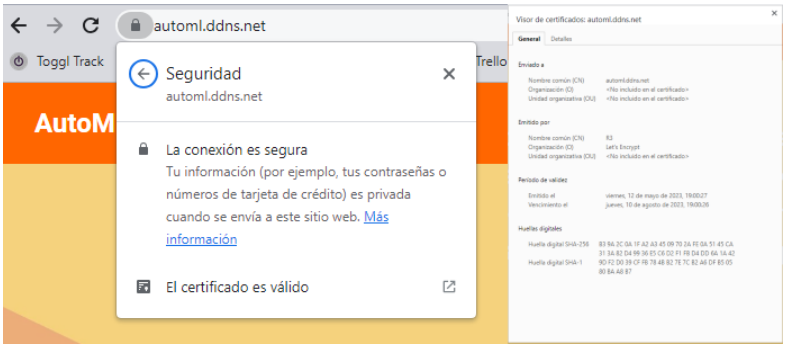

Figura 4.2: Información sobre la validez de los certificados generados para `https://automl.ddns.net`

En el anexo A.1.4 se puede obtener más información sobre el proceso de obtención de los certificados.

4.1.1. Google Cloud Platform (GCP)

Una vez determinada la arquitectura de nuestra solución necesitamos un entorno dónde desplegarlo que tenga las siguientes características:

- Instalación de *Docker*
- Disponibilidad *24/7*
- IP estática
- Escalabilidad
- Configuración de Firewall

Teniendo en cuenta todas estas características y, la necesidad de entrar en un presupuesto reducido, se ha tomado la decisión de utilizar una instancia de máquina virtual de Google Cloud Platform (GCP).

GCP es una plataforma en la nube de Google que ofrece recursos y herramientas para la implementación y gestión de máquinas virtuales (MV). Permite a los usuarios crear y ejecutar MV en la nube de forma simple, escalable y flexible.

Para este proyecto se ha configurado una máquina virtual con estos componentes:

- Zona: Europa (Para reducir la latencia)
- Plataforma: e2-standard-2
- Disco: 50 GB ampliables
- Sistema operativo: Ubuntu 20.04

- IP estática

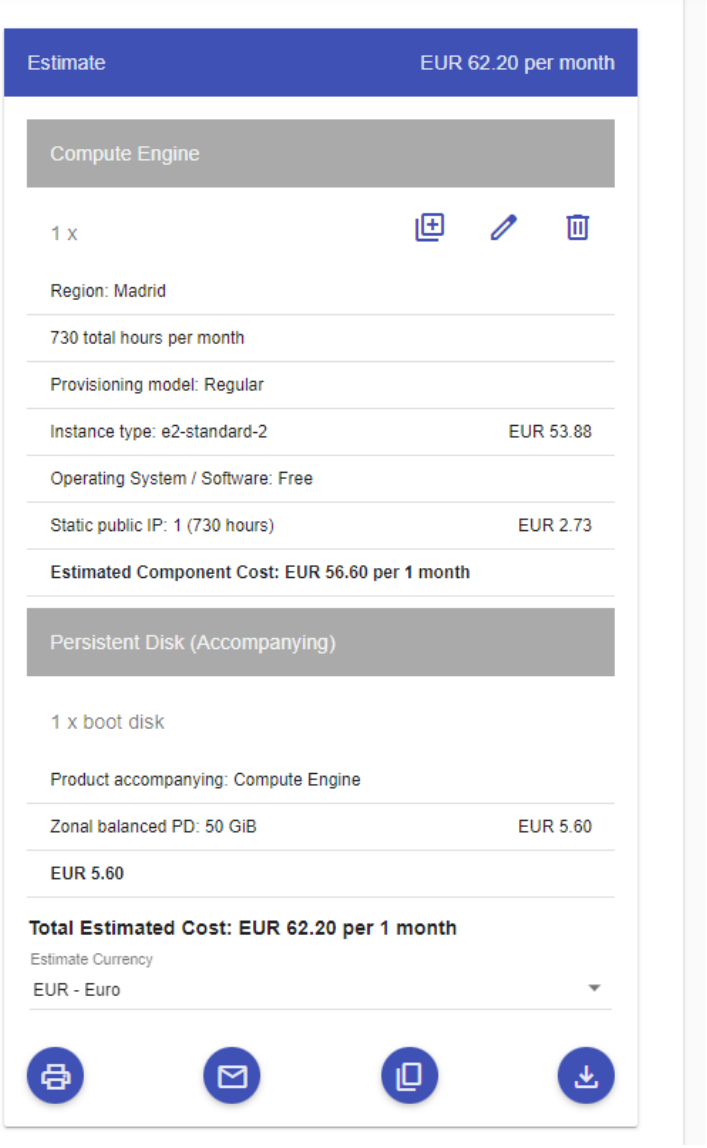

Figura 4.3: Estimación de costes de la máquina virtual configurada

En caso de encontrarse encendida *24/7* el coste mensual de la máquina se encuentra entorno a los 62.20 € (los precios pueden variar ligeramente con el tiempo). Para poder tener la información actualizada de los precios se puede acceder a [9]

En el anexo A.1.1 se entra en profundidad de la creación de la máquina, configuración del Firewall, instalación de *Docker* y otras configuraciones tenidas en

cuenta.

4.2. Aplicación web

4.2.1. FastAPI

FastAPI es un framework de desarrollo web en Python diseñado específicamente para la creación rápida de APIs. Se utiliza ampliamente para desarrollar servicios web y aplicaciones que requieren una interacción eficiente entre el frontend (Flask) y el backend (AutoML).

En nuestro caso se ha desarrollado una API con múltiples endpoints. Estos a su vez se pueden categorizar en **4** grandes grupos:

- Interacción con el usuario
- Funciones de Dataset
- Uso de AutoML
- Funciones comunes

Estos endpoints serán utilizados como una API privada. Esto implica que los detalles del funcionamiento no son relevantes para el usuario final de la aplicación, sino que solamente son útiles para los desarrolladores.

Dataset

Dataset es uno de los puntos principales de la librería y por ello es necesario permitir la interacción mediante endpoints. Para el diseño de las funcionalidades se ha seguido el patrón CRUD[2]:

- Create (crear): Es posible generar un Dataset utilizando el fichero y añadiendo un JSON con la configuración necesaria.
- Read (leer): Haciendo uso del proceso de autenticación descrito en 4.4, se puede acceder a la información de los Datasets creados por el usuario.
- Update (actualizar): Es posible modificar atributos del Dataset como el tipo de dato inferido para cada variable.
- Delete (eliminar): Existe la posibilidad de eliminar un Dataset, siempre y cuando se tengan los permisos necesarios.

[2]CRUD: Create, Read, Update and Delete. Es un acrónimo para las operaciones que se pueden realizar sobre información almacenada[5]

AutoML

De manera similar a Dataset se han implementado las funcionalidades CRUD para la creación y modificación de los experimentos creados por el usuario.

A mayores, los experimentos necesitan tener más funcionalidades cómo ser entrenados y hacer predicciones. Para ello se han generado dos endpoints específicos para cada una de estas tareas.

- Entrenamiento: Dado el identificador del experimento y, verificando los permisos del usuario, se lanza el entrenamiento de todas las *pipelines* generadas por *OrangePrint* (3.3.1)
- Predicción: Cada *pipeline* tiene un identificador único dentro del experimento. Haciendo uso de este identificador y subiendo un fichero con los nuevos datos se realiza la predicción y se descarga automáticamente un fichero con los resultados.

Funcionalidades comunes

Durante el desarrollo de la API se han detectado una serie de funcionalidades que se utilizan de manera recurrente por múltiples endpoints. Por ello, se ha decidido extraer dichas funcionalidades a endpoints propios los cuales pueden ser llamados.

Principalmente estas funcionalidades están relacionadas con la autorización de los usuarios:

- check_token: Verifica que el token proporcionado por el usuario es válido. Esto permite dar acceso a cada usuario únicamente a la información de sus Datasets y experimentos.
- check_admin: Una capa por encima del anterior que además comprueba si los permisos del usuario son de tipo administrador.

Interacción con el usuario

Este conjunto de *endpoints* de la API se encarga de la gestión de usuarios. Su función principal es servir como intermediario entre la aplicación web y la base de datos. Las funcionalidades que implementan son:

- Registro
- Inicio de sesión
- Cierre de sesión

En este punto es importante recalcar un problema de ciberseguridad. La base de datos es un punto frágil del sistema y pese a todos los intentos por evitar ser penetrada puede llegar a ocurrir. Por ello, es fundamental no almacenar información confidencial como las contraseñas en texto plano.

En está aplicación se ha tomado la decisión de pasar las contraseñas por un algoritmo de *hashing* irreversible. A continuación se muestra un diagrama de como funcionaría el proceso de registro, inicio de sesión y solicitud a un *endpoint* protegido:

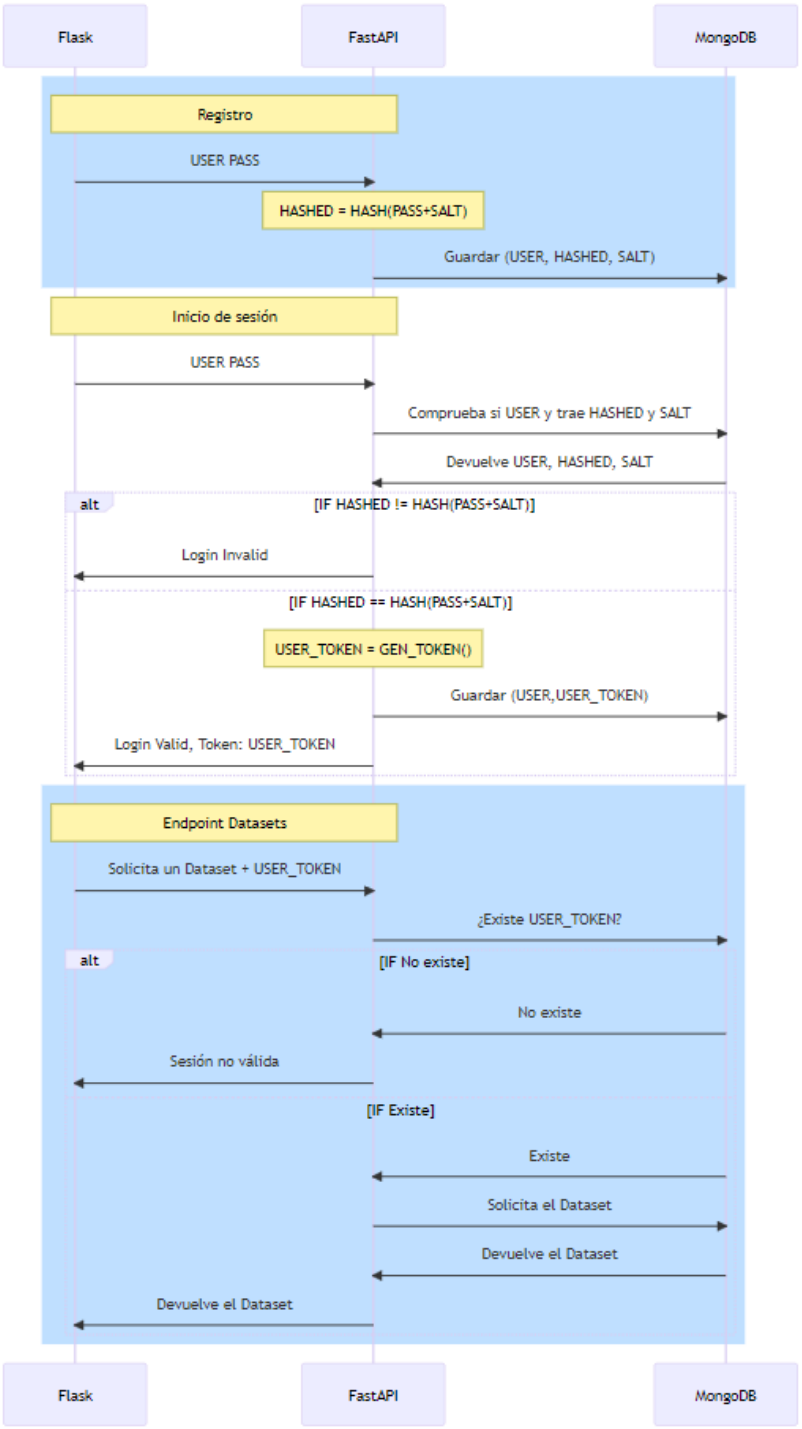

Figura 4.4: Flujo de acceso a endpoints protegidos de la aplicación

La metodología de uso de funciones de *hash* con una palabra secreta (*salt*) sigue el patrón descrito en [15]. Gracias a este diseño en caso de lograr acceso a la base de datos la cantidad de información personal filtrada se minimiza.

La implementación técnica y mayores detalles sobre cada uno de estos endpoints se puede leer en B.1. En caso de ser administrador de la web y por tanto, necesitar conocimientos profundos sobre estos servicios se puede acceder a una documentación interactiva en `https://automl.ddns.net/admin/api/docs`

4.2.2. Flask

Este servicio es el único punto con el que el usuario interacciona directamente. Para su desarrollo ha sido necesario conocimientos tanto en HTML como en CSS para la generación de la interfaz de usuario.

Se ha utilizado un esquema de tipo estático, es decir, cada una de las páginas disponibles en la aplicación es un fichero HTML diferente; en lugar de cargar de forma dinámica sobre un único fichero. Esto ha sido debido a las limitaciones de *Flask* y el *framework* de *Django* que utiliza para renderizar los ficheros HTML.

Otro punto a destacar es que *Flask* es totalmente independiente de la implementación de la librería AutoML. De hecho, *FastAPI* actúa como interfaz entre ambos haciendo que los cambios en AutoML solamente se vean reflejados en el diseño de *FastAPI*.

De cara a presentar las diversas funcionalidades desarrolladas se hará el flujo completo de: registrarse, iniciar sesión, subir un fichero, analizarlo, crear el experimento, entrenar y realizar una predicción.

Página de inicio

Al entrar en `https://automl.ddns.net` se muestra la siguiente web en la que se permite ir a la sección de registro o incio de sesión en caso de tener un usuario.

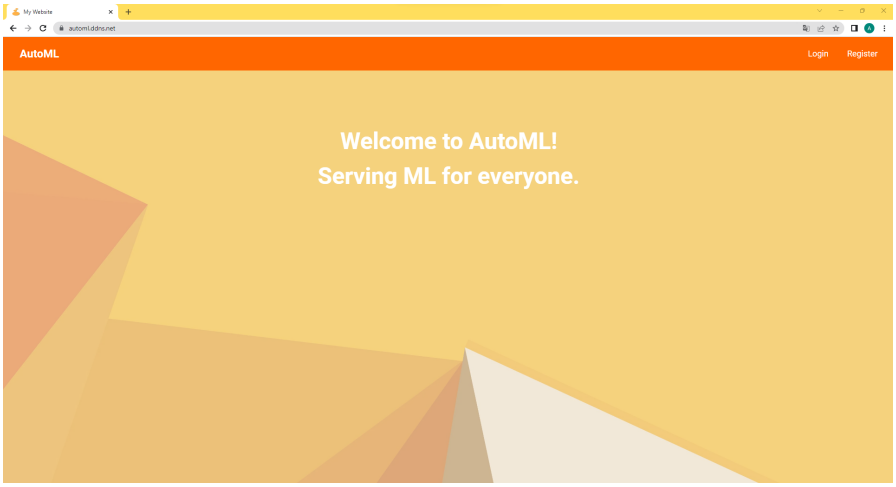

Figura 4.5: Página de llegada de la web

Registro de usuario

Al hacer click en el botón *Register* se redirige a:

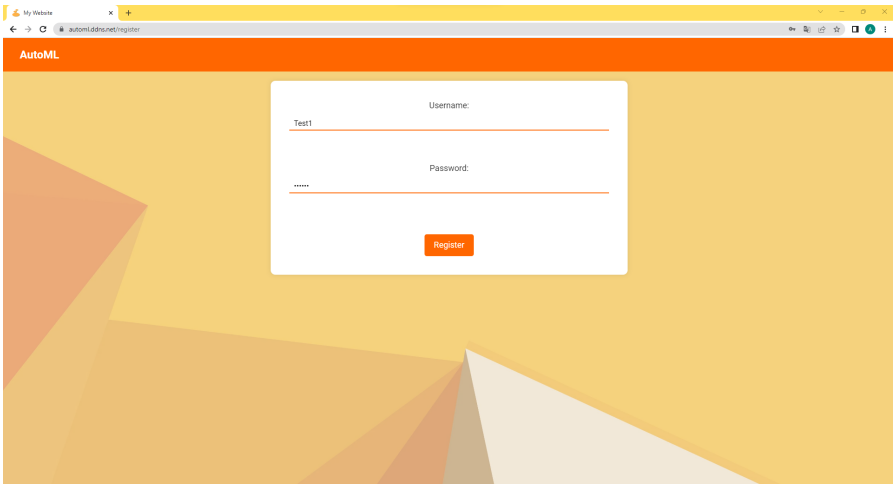

Figura 4.6: Página de registro

Durante el registro se verifica que el nombre de usuario este disponible y que la contraseña cumple con unos estándares mínimos. Una vez registrado se redirige automáticamente a la página de inicio de sesión.

Inicio de sesión

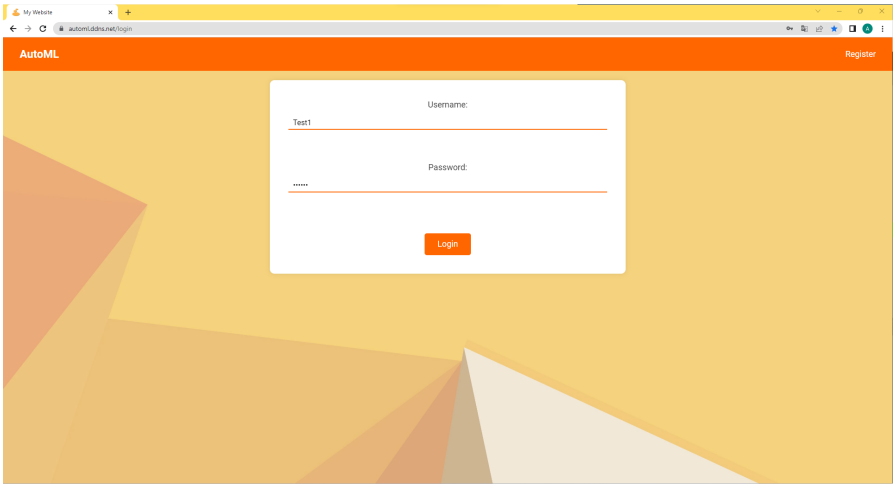

Figura 4.7: Inicio de sesión

Se solicita al servicio de *FastAPI* la validez del usuario haciendo uso del esquema descrito en 4.4. En caso de ser válido el usuario llegará a su Dashboard.

Dashboard

En la web del *Dashboard* se muestran todos los Datasets y experimentos con el usuario como propietario. En caso de ser un usuario recién registrado se verá lo siguiente:

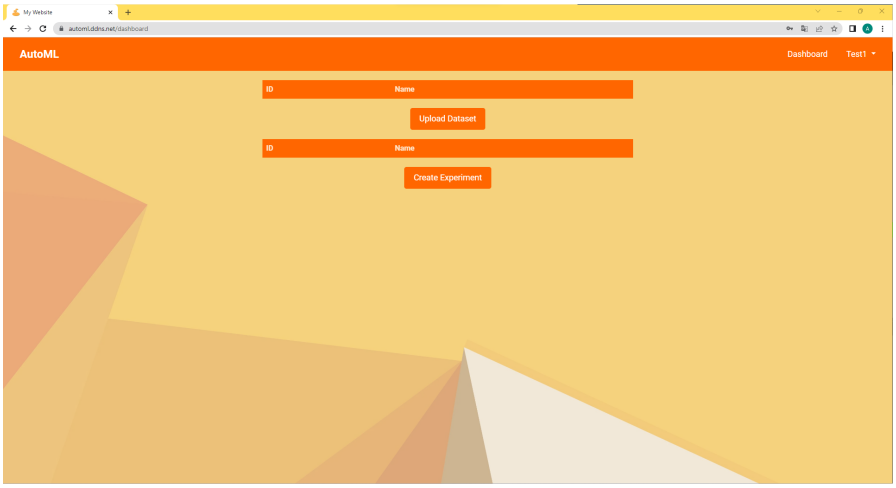

Figura 4.8: Dashboard de un usuario nuevo

Para subir un fichero basta con hacer click en *Upload Dataset*.

Subida de un Dataset

La subida de un Dataset se separa en dos partes. Inicialmente, se solicita subir el fichero e indicar el separador. En este momento, *Flask* verifica que el fichero tiene un formato válido e identifica las múltiples columnas.

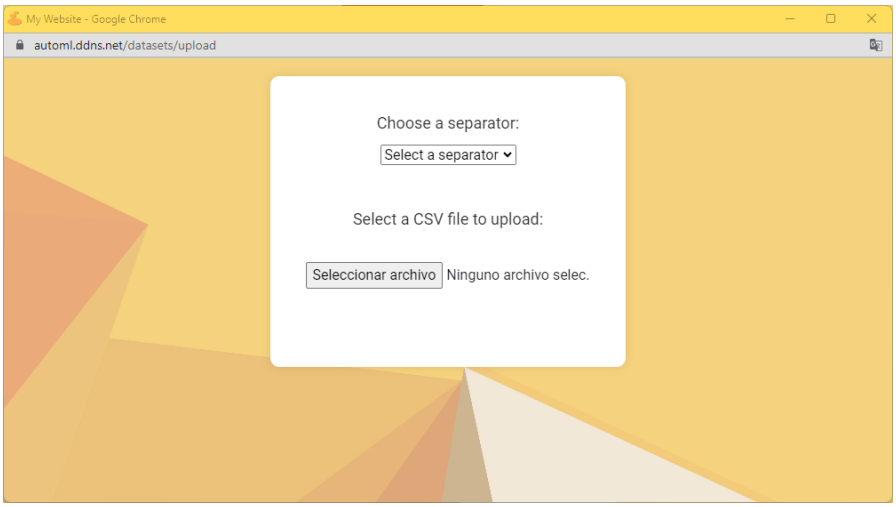

Figura 4.9: Subida de fichero a AutoML

Una vez todas las verificaciones se han llevado a cabo se le solicita al usuario poner un nombre (por defecto utiliza el nombre del fichero sin .csv) y seleccionar cual es la columna a predecir.

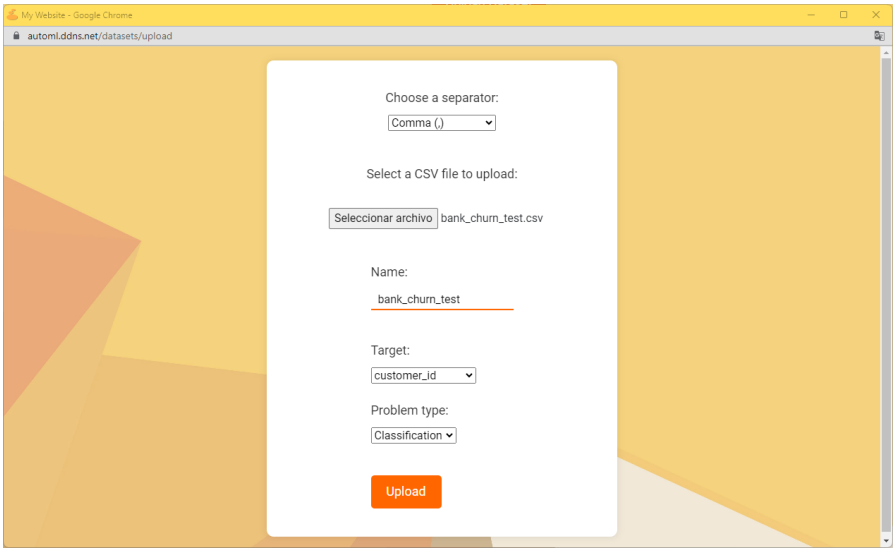

Figura 4.10: Ajustes en la subida del fichero

Al finalizar se cierra la ventana y volvemos al dashboard actualizado. Además, ahora aparecen tres nuevas funcionalidades: *View*, *Download*, *Delete*.

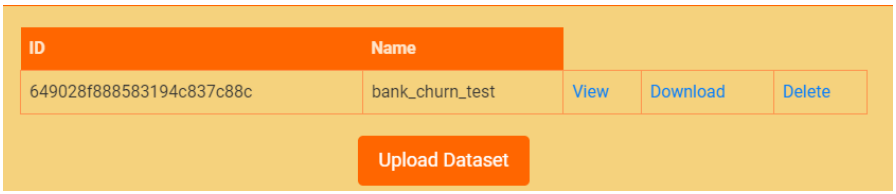

Figura 4.11: Dashboard con Dataset subido

Información del Dataset

Figura 4.12: Información del Dataset

Podemos visualizar información básica sobre el Dataset como el número de nulos, valores más frecuentes, etc. También se implementan otras dos posibilidades:

- Modificar el tipo de dato inferido
- Generar visualización

Para la generación de la visualización se hace uso de la librería *SweetViz* la cual, hace un análisis exploratorio inicial del Dataset.

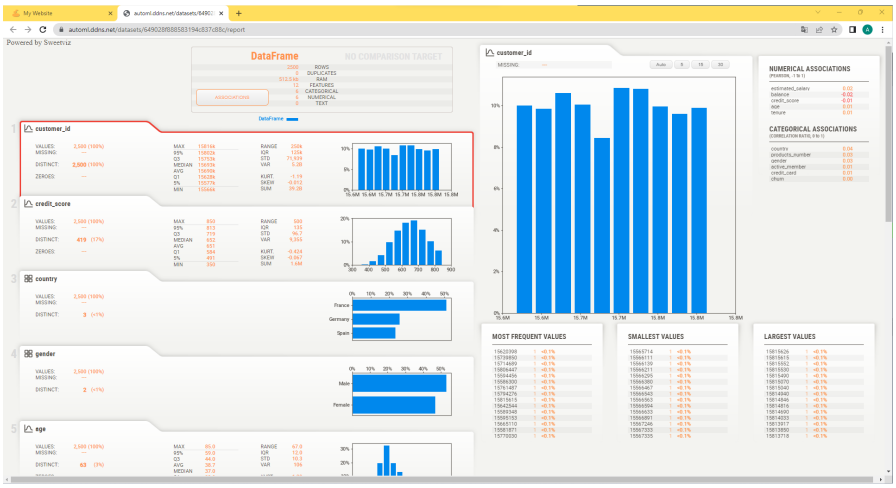

Figura 4.13: Visualización generada con SweetViz

Creación del experimento

Para la creación de un experimento desde el Dashboard se abre una nueva ventana dónde se despliegan los Datasets disponibles. Una vez seleccionado basta con asignar un nombre al experimento y ya podemos entrenar.

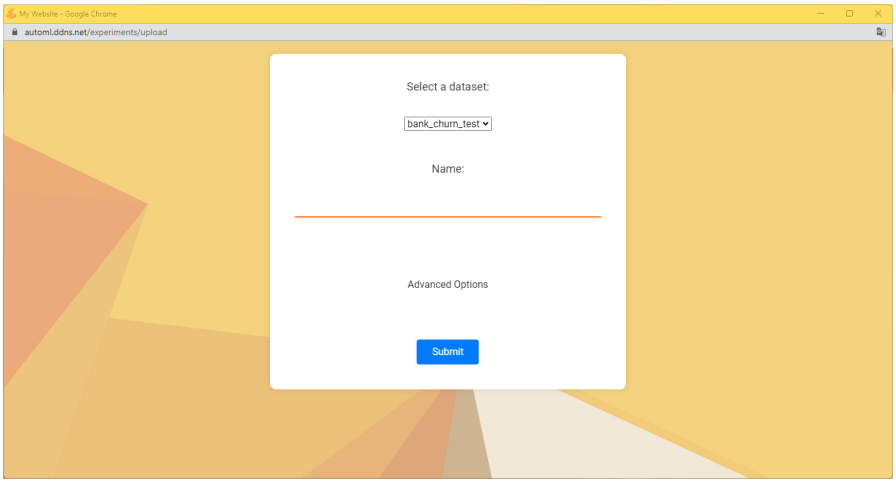

Figura 4.14: Creación por defecto de un experimento

En caso de querer ajustar que tipo de modelos o preprocesados se desea utilizar para el experimento, se puede ir a los ajustes avanzados y marcar para cada categoría los valores deseados. Estos serán pasados en formato JSON como configuración a *Orangeprint* 3.3.1

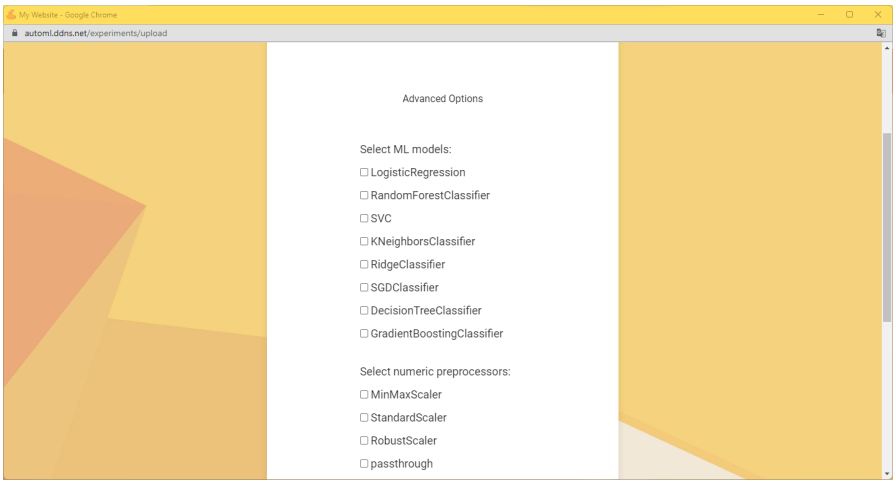

Figura 4.15: Opciones avanzadas para la creación de un experimento

De manera similar a los Datasets, una vez creado se cierra la ventana emergente

y se actualiza automáticamente el Dashboard.

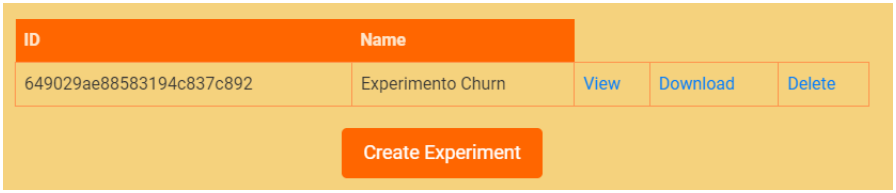

Figura 4.16: Experimento en el Dashboard

Entrenamiento de los modelos

Una vez creados los experimentos se disponibilizan al usuario los modelos que ha generado el módulo de *OrangePrint*. Dentro de está página se encuentra el botón *Train* que inicializa el entrenamiento de todos los modelos.

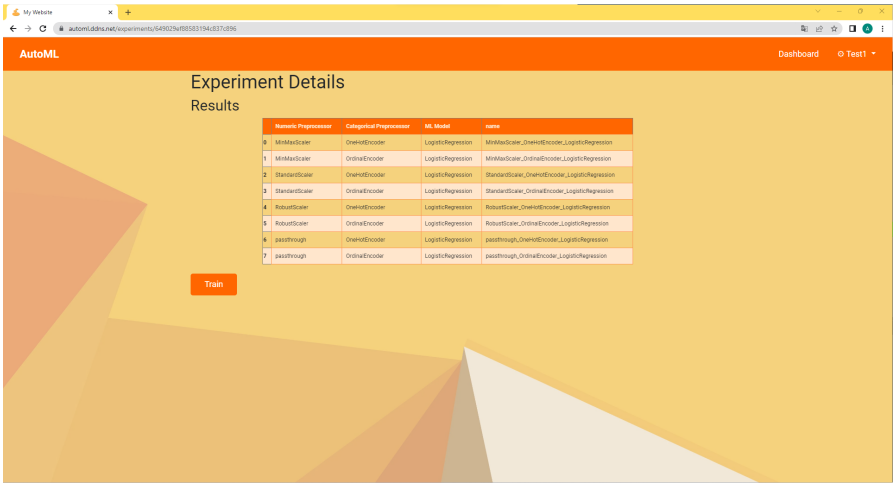

Figura 4.17: Información sobre el experimento creado.

Al finalizar el entrenamiento la página se actualiza y muestra los resultados de cada uno de los modelos, aquí el usuario puede elegir que modelo utilizar para las predicciones en función de su necesidad y las métricas.

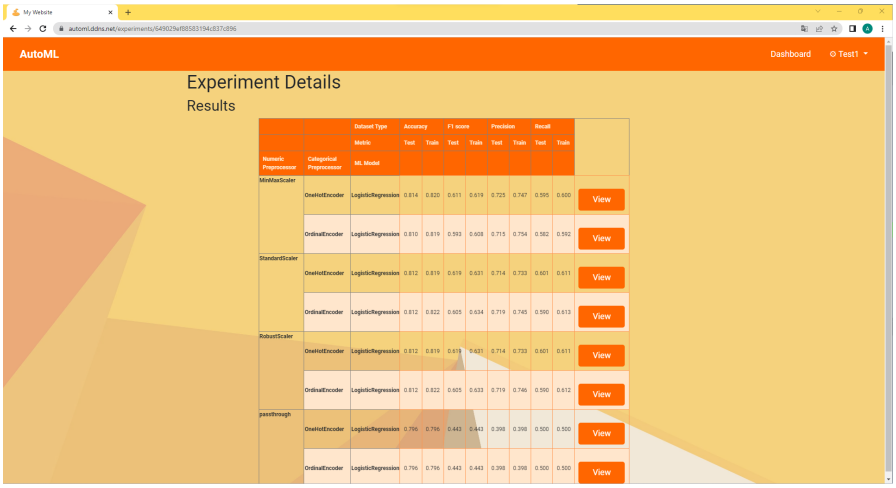

Figura 4.18: Resultados del entrenamiento de los modelos

Predicción

Para cada modelo entrenado se disponibiliza una *url* en la que se puede ver el diseño como un gráfico y realizar las predicciones sobre un conjunto nuevo de datos con la misma estructura que el Dataset con el que se ha entrenado.

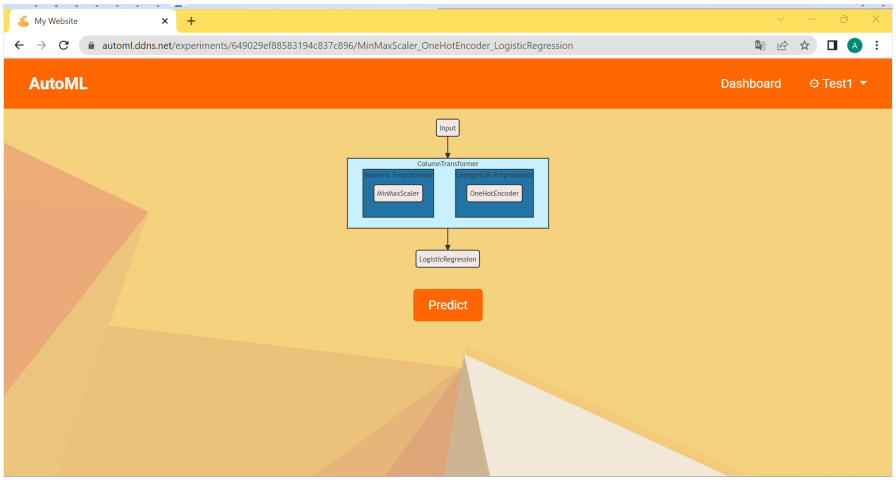

Figura 4.19: Información sobre cada modelo entrenado.

En un futuro se pueden añadir otras funcionalidades en está página como la importancia de variables, hiperparámetros del modelo, métricas avanzadas, etc. De momento por limitar el alcance del proyecto la única posibilidad es realizar predicciones.

Para ello el proceso es similar al de 4.9 sólo que en este caso solamente se solicita subir el fichero y el separador. Una vez se ha subido la aplicación devolverá el mismo fichero pero con una nueva columna con la predicción.

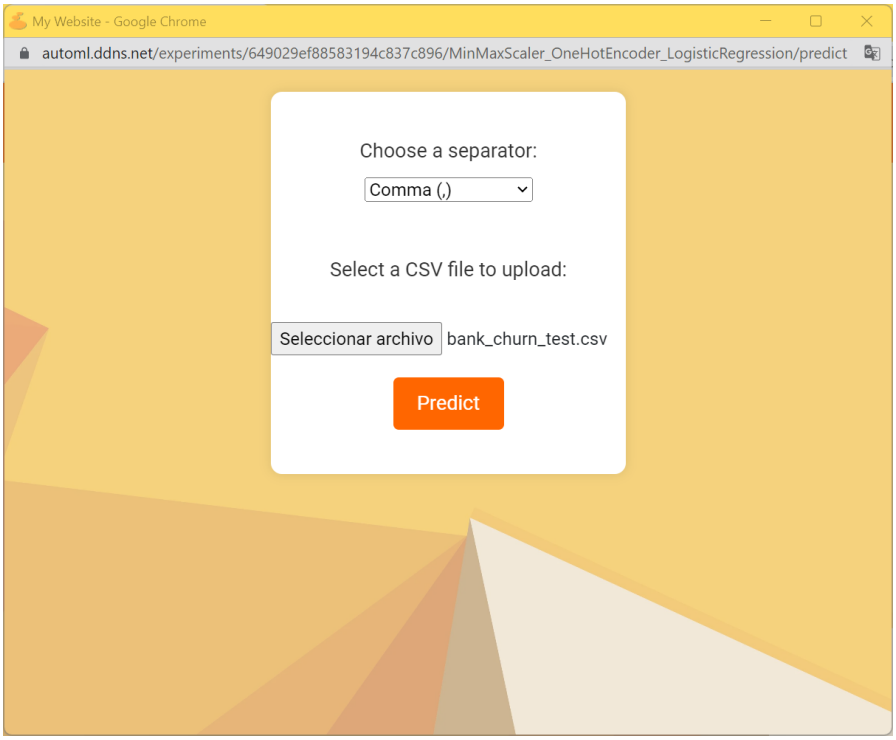

Figura 4.20: Predicción de un nuevo conjunto de datos

Por ultimo, también se ha creado una versión un poco diferente en caso de ser un usuario de tipo administrador. Este tendrá acceso al panel de administrador con la documentación de la API de *FastAPI* y a la herramienta de *Mongo-Express*.

Figura 4.21: Panel de administrador.

Con esta guía de un flujo de uso completo hemos pasado por la gran mayoría de funcionalidades implementadas en la página web. Para mayor detalle se puede consultar el anexo B.2

4.2.3. MongoDB

MongoDB es una popular base de datos NoSQL que almacena datos en formato BSON, ofreciendo alta escalabilidad y rendimiento rápido. Además de su capacidad para gestionar datos estructurados y semiestructurados, MongoDB también ofrece GridFS, un sistema de almacenamiento en el que se pueden guardar y recuperar archivos de gran tamaño de forma eficiente.

GridFS es útil cuando se necesita almacenar archivos más grandes de lo que puede soportar el modelo de documentos BSON tradicional de MongoDB. En lugar de almacenar un archivo completo en un solo documento, GridFS divide el archivo en fragmentos más pequeños y los almacena en colecciones separadas, lo que facilita su manejo y recuperación.

Al utilizar GridFS, es posible almacenar archivos de cualquier tamaño en MongoDB y acceder a ellos fácilmente mediante consultas y operaciones de lectura/escritura. Esto hace que sea de gran utilidad para nuestra aplicación por la posibilidad de poder guardar los Datasets y experimentos de manera cómoda y rápida.

Se ha creado una base de datos llamada AutoML y dentro de ellas las siguientes colecciones (equivalentes a tablas en las bases de datos relacionales):

- Datasets: Contiene la *metainformación* de cada uno de los Datasets subidos a la plataforma (ID de fichero en GridFS, ID del usuario propietario y nombre del fichero)
- DataViz: Las visualizaciones generadas mediante *SweetViz* quedan almacenadas a modo de caché para evitar regenerarlas.
- Experiments: Similar a Datasets contiene la información necesaria para identificar un experimento.
- Sessions: Los tokens generados en el diagrama 4.4 se almacenan en esta colección. La duración de los tokens está configurada a un día o al cierre de sesión del usuario.
- Users: Información sobre los usuarios.

Debido al uso de GridFS también se han generado de manera automática las siguientes colecciones:

- fs.chunks: Contiene los ficheros divididos en múltiples *chunks*
- fs.files: Información sobre los ficheros como número de *chunks* y tamaño.

Para facilitar la visualización de los datos, se han disponibilizado en `https://automl.ddns.net/admin/mongo-express` con la herramienta *Mongo-express*. Dado que la herramienta da permisos de administrador sobre la base de datos se ha limitado el acceso mediante dos métodos complementarios.

- Filtro de conexiones por IP
- Usuario y contraseña

Este nivel de seguridad se ha tomado debido a una vulnerabilidad detectada durante la fase de desarrollo que permitió a unos atacantes obtener toda la información de la base de datos y solicitar un pago a cambio de recuperar la información.

Por suerte, gracias a las medidas de prevención tomadas para la información de los usuarios, la filtración de datos no llego a un problema más grave que restaurar la copia de seguridad generada cada noche de forma automática.

4.3. Ciberseguridad

A continuación, se hace una recopilación de todas las medidas de seguridad implementadas durante el despliegue de la aplicación web ordenadas de mayor a menor importancia.[13]

- Certificado HTTPS: El uso de un certificado HTTPS garantiza una conexión segura entre el cliente y el servidor, cifrando la información transmitida y verificando la autenticidad del sitio web.
- Configuración del Firewall: El firewall se configura para controlar el tráfico de red y proteger el servidor contra posibles ataques. Se pueden establecer reglas para permitir o bloquear determinados tipos de conexiones, puertos o direcciones IP, con el fin de limitar las vulnerabilidades y garantizar la seguridad del sistema. Por ejemplo, es habitual bloquear las conexiones SSH al servidor para ser solo válidas desde una lista reducidad de IPs.
- Protección de la base de datos: Se establecen credenciales de acceso seguras para la administración de la base de datos. Además, se limita el acceso solo a determinadas direcciones IP autorizadas, lo que añade una capa adicional de protección.
- Hashing de contraseñas con salt: Las contraseñas de los usuarios se almacenan en forma de hash, lo que garantiza que las contraseñas originales no sean accesibles incluso si la base de datos se ve comprometida. El *salt* se añade al proceso de *hash* para hacer más difícil la decodificación mediante ataques de fuerza bruta.
- Tokens de autenticación: Se utilizan tokens para validar la identidad de los usuarios y autorizar sus acciones en la aplicación. Estos tokens se generan durante el inicio de sesión exitoso y se incluyen en las solicitudes posteriores para verificar la autenticidad del usuario.

- Proxy inverso para ocultar ubicaciones del servidor: Se utiliza un proxy inverso para actuar como intermediario entre los clientes y el servidor real. Esto ayuda a ocultar la ubicación exacta del servidor y proporciona una capa adicional de seguridad al filtrar y controlar el tráfico de red.

Capítulo 5

Conclusiones y trabajos futuros

En este capítulo se describen las conclusiones obtenidas, junto con los posibles trabajos futuros que se han identificado durante el desarrollo del proyecto. Para ello, se analizará el cumplimiento de los objetivos planteados al inicio.

El objetivo principal del proyecto de **desarrollar una aplicación que permitiera el desarrollo de modelos de aprendizaje automático para clasificación y su análisis en la nube** ha sido satisfecho, ya que no solamente se ha desarrollado dicha herramienta sino que se ha creado una aplicación web que permite desplegar esta herramienta como un servicio para múltiples usuarios.

Desglosando por partes el objetivo principal, tenemos que se ha cumplido lo siguiente:

- **Automatización del análisis exploratorio de los datos**: Se ha satisfecho mediante el uso de la clase Dataset con ayuda de la librería *SweetViz*.
- **Automatización de la creación de modelos de aprendizaje automático**: Cumplido mediante la implementación de la clase *OrangePrint* que permite la generación y optimización de modelos dado un problema de clasificación.
- **Despliegue y puesta en producción de dichos modelos.**: La aplicación web permite realizar predicciones sobre un modelo entrenado para ficheros nuevos de datos.
- **Facilitar la gobernanza de modelos de *Machine Learning.***: Todo el control sobre los datos utilizados para el entrenamiento y los modelos se encuentra centralizado en una única aplicación.

A continuación, se hará una recopilación de las dificultades encontradas durante el desarrollo de la web, además de exponer problemas actuales y las posibles soluciones a futuro.

5.1. Dificultades encontradas

El diseño de la librería de Python no tuvo grandes dificultades ya que solamente requería de conocimientos en programación orientada a objetos y de librerías de Python muy utilizadas como *scikit-learn* o *Pandas*.

Las dificultades surgieron cuando se inicio el desarrollo de la interfaz web. Para ello se ha llevado a cabo una primera investigación de que herramientas serían necesarias para construir la aplicación web; lo que derivó en la necesidad de documentarse en lo siguiente:

- **HTML**: Lenguaje necesario para dar estructura a la página web.
- **JavaScript**: Lenguaje que permite interactuar a una página web con los endpoints diseñados.
- **CSS**: Configuración de estilos de la web.
- **Dockers**: Configuración de dockers para la virtualización de microservicios.
- **Arquitectura de red**: Todos los problemas que conlleva configurar puertos, redes, permisos, etc. 4.1
- **Diseño de APIs con FastAPI** 4.2.1
- **Diseño de endpoints web con Flask** 4.2.2
- **Google Cloud Platform** 4.1.1
- **Ciberseguridad** 4.3

5.2. Trabajos futuros

Todos los proyectos deben tener definido claramente el alcance para evitar que se prolonguen en el tiempo. Es por ello, que para está primera versión de la aplicación han quedado muchas funcionalidades fuera del alcance y se proponen para futuros desarrollos.

Estas tareas se pueden separar en dos tipos: problemas de arquitectura y funcionalidades de la librería de AutoML. Los primeros son los más fundamentales a resolver en caso de poner en producción la aplicación web debido a las limitaciones actuales en caso de aumentar el número de usuarios. Todos estos problemas se resumen en la siguiente tabla:

Tipo	**Problema**	**Descripción**	**Solución**	Importancia
Arquitectura	Paralelización de tareas	Actualmente el servidor no está preparado para gestionar múltiples peticiones de alta carga computacional de manera paralela.	Implementar un sistema de gestión de tareas mediante Celery y REDIS.	Muy alta
AutoML	Optimización hiperparámetros	Debido al problema anterior, los modelos no realizan una búsqueda profunda de los mejores hiperparámetros para cada problema.	Una vez arreglado la gestión de tareas, implementar métodos de búsqueda para cada modelo y experimento.	Alta
AutoML	Problemas de regresión	No están disponibles las soluciones a los problemas de regresión.	Realizar la implementación necesaria en AutoML.	Media
AutoML	Añadir series temporales.	No permite resolver problemas de series temporales.	Implementar en la librería de AutoML las series temporales.	Media
AutoML	Baja cantidad de preprocesados	En la versión de la herramienta el número de preprocesados que realiza es mínimo. Esto es especialmente relevante en caso de querer implementar los problemas de series temporales.	Añadir módulo de ingeniería de variables para mejorar el preprocesamiento de los datos.	Baja

Cuadro 5.1: Puntos débiles y futuras mejoras de la aplicación de AutoML

Apéndice A

Manual de instalación

El código de la aplicación se ha desarrollado principalmente haciendo uso de Python y de Git como herramienta de control de versiones. Como servicio de alojamiento del repositorio de Git se ha hecho uso de *Github*. Actualmente, se encuentra como repositorio privado por motivos de licenciamiento del código pero, se puede solicitar acceso de lectura al repositorio enviando un correo electrónico a 202207308@alu.comillas.edu e indicando el nombre de usuario de *Github*.

A.1. Despliegue de la aplicación

Para el presente proyecto se ha decidido hacer uso de una plataforma de computación en la nube debido a la capacidad de estar disponible *24/7* y la facilidad de asignar un dominio manteniendo la IP estática. Pese a ello, la configuración se puede llevar a cabo siempre y cuando se disponga de una máquina con el sistema operativo Linux, con acceso a internet y una IP estática. El manual de usuario para este segundo caso se encuentra en la sección A.1.2.

A.1.1. Creación máquina virtual en Google Cloud Platform (GCP)

La plataforma de computación en la nube seleccionada ha sido GCP, lo primero que se necesita es disponer de una cuenta con un método de pago asociado. Durante los primeros 90 días a Junio de 2023 GCP regala 300$ en créditos para realizar pruebas en la plataforma.

Una vez creada la cuenta nos llevará a la consola de GCP y crearemos una máquina haciendo click en 'Crear una VM', tal y como se muestra en la Figura A.1:

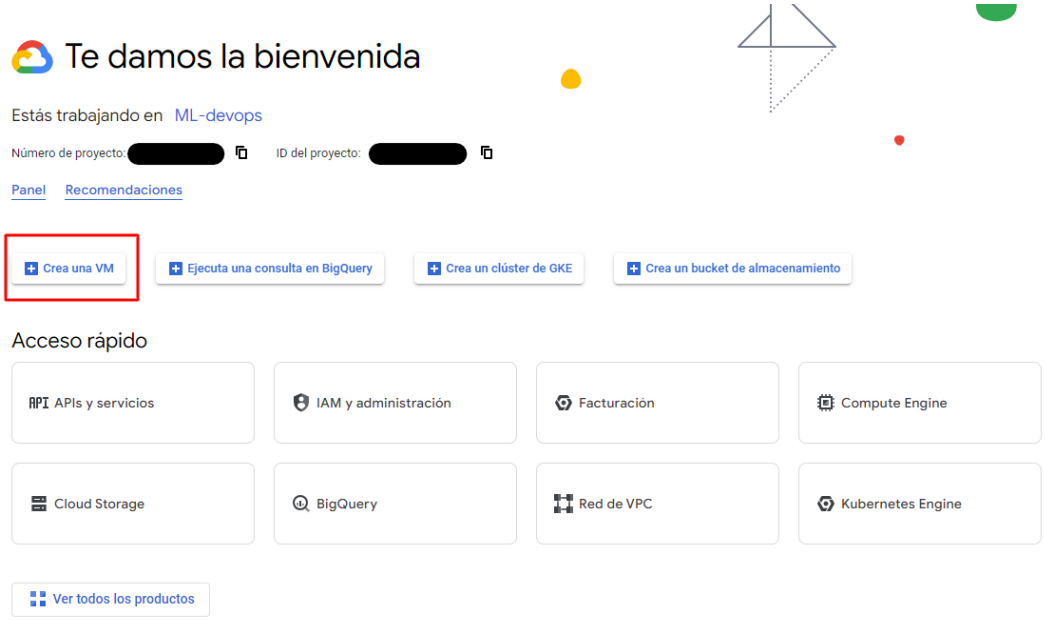

Figura A.1: Consola de GCP

En esta sección, se podrá configurar la máquina virtual con los recursos que sean necesarios. En caso de no tener los créditos gratuitos GCP ofrece la plataforma *e2-micro* de forma gratuita a cambio de un menor rendimiento. Para el proyecto se ha utilizado la siguiente configuración:

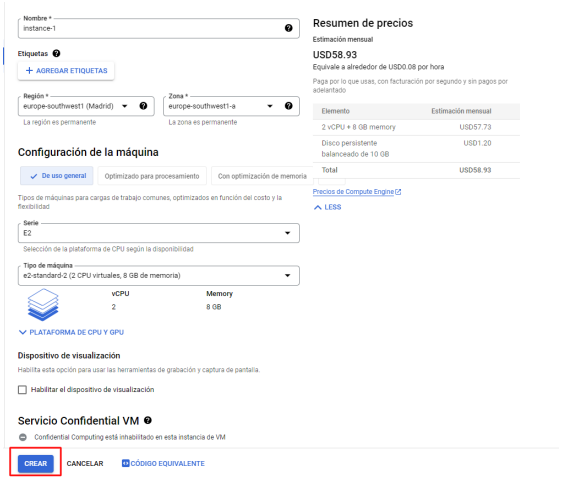

Figura A.2: Configuración de la máquina virtual

Una vez creada se tendrá acceso a los detalles de la máquina desde la sección del *Compute Engine* (accesible desde la consola), se podrá ir a la configuración de red

para asignar una IP pública estática y configurar el firewall.

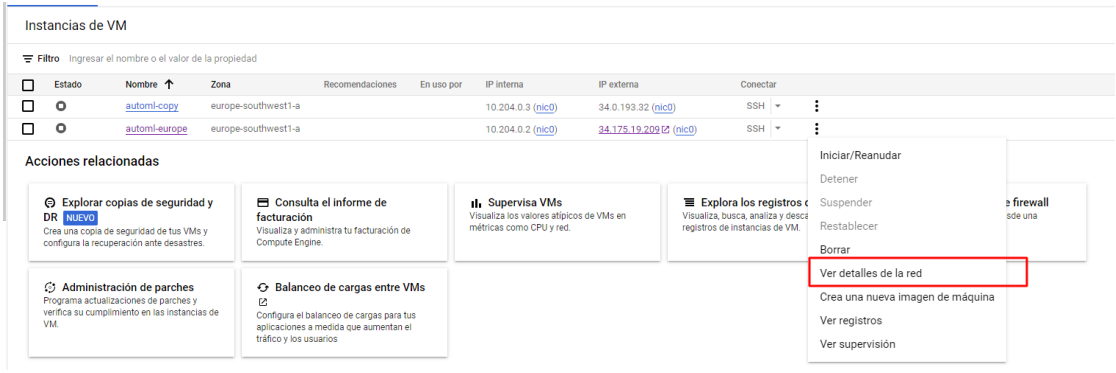

Figura A.3: Acceso a configuración de red

Para la IP estática se tiene que hacer *click* en 'Direcciones IP' , dentro en 'Reservar dirección IP externa estática' y seguir los pasos que se indiquen.

Figura A.4: Configuración IP estática

Para el firewall se dejarán abiertos los puertos 443 (HTTPS) y 80 (HTTP) para todas las IPs (0.0.0.0/0). Momentáneamente para configurar nuestro sistema dejaremos abierto el puerto 22 a todas las IPs, aunque debemos recordar configurarlo para funcionar sólo desde una lista de IPs restringidas (IP del domicilio, trabajo, VPN, etc.).

Ahora que ya se dispone de IP estática y se ha configurado el firewall se puede proceder a obtener un dominio gratuito con la pagina web `https://www.noip.com/es-MX`. En esta parte se crea un nuevo host con la IP asignada por GCP en el paso anterior.

En este punto, se dispone de una máquina virtual con una IP estática y un dominio para facilitar el acceso sin recordar la IP. Los próximos pasos consisten en configurar el sistema para conectarse a la máquina virtual con facilidad.

Para ello se debe configurar un usuario con permisos de root en la máquina y configurar las claves SSH. Inicialmente, se necesita utilizar la interfaz de GCP para hacer SSH desde el *Compute Engine*.

Figura A.5: Acceso por SSH

Se crea un nuevo usuario y le asignamos derecho de administrador haciendo uso de los siguientes comandos:

```
g202207308@automl-copy:~$ sudo adduser testing
Adding user `testing' ...
Adding new group `testing' (1003) ...
Adding new user `testing' (1002) with group `testing' ...
Creating home directory `/home/testing' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for testing
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n]
g202207308@automl-copy:~$ sudo adduser testing sudo
Adding user `testing' to group `sudo' ...
Adding user testing to group sudo
Done.
g202207308@automl-copy:~$
```

Figura A.6: Creación de usuario con permisos de administrador

En caso de no tener claves SSH, se debe crear unas claves SSH propias, esto se puede realizar desde el terminal del ordenador propio (**no** la máquina virtual), escribiendo el comando `ssh-keygen`. Esto generará un par de claves pública-privada en el directorio `C:/Users/NOMBRE_DE_USUARIO/.ssh` .

Se procede a copiar el contenido de la clave pública (fichero que termina en .pub) para introducirlo en un fichero especifico de la máquina de Google mediante los siguientes comandos:

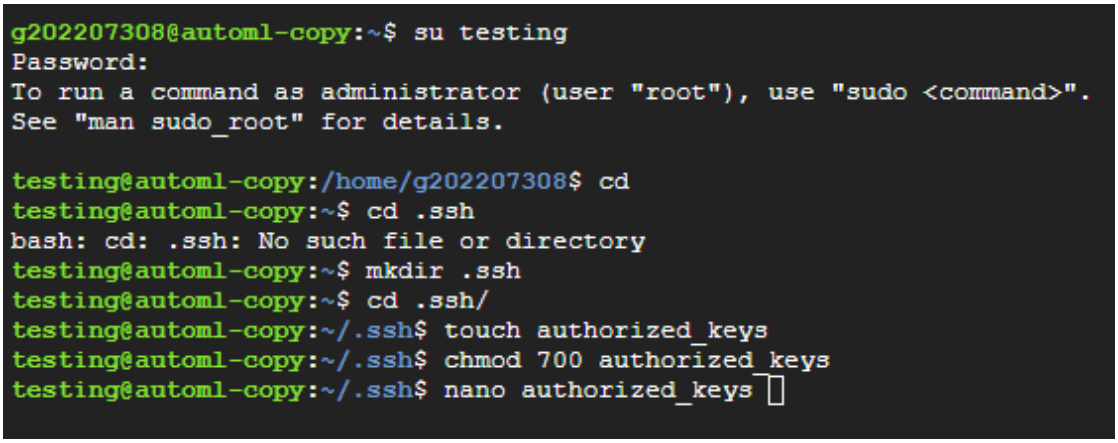

Figura A.7: Añadir claves SSH en el servidor.

Con el comando `nano` se edita el fichero añadiendo la clave pública generada y guardando con `Ctrl+x`. Una vez realizado este paso, se puede cerrar la ventana de Google y revertir el cambio temporal en el Firewall para el puerto 22.

Para configurar la conexión desde nuestro equipo se modificará el fichero *config* que se encuentra en `C:/Users/NOMBRE_DE_USUARIO/.ssh` (en caso de no existir se crea uno sin ninguna extensión) y se pega dentro tal y como se detalla a continuación:

```
Host ginstance
     HostName IPESTATICA
     User testing
     Port 22
     IdentityFile ~/.ssh/id_rsa
     IdentitiesOnly yes
```

Con esto ya configurado se puede proceder a la conexión a la máquina con el comando *ssh ginstance* en el terminal de windows.

En caso de utilizar Visual Studio Code podremos programar en la máquina virtual como si fuera en local haciendo uso de la extensión para SSH de Visual Studio Code que detectará automáticamente los servidores configurados en el fichero *config*.

De aquí en adelante se supondrá que estamos utilizando Visual Studio Code y tenemos configurado la extensión de SSH.

A.1.2. Instalación de Docker y docker-compose

A continuación se describen los pasos llevados a cabo para la instalación de Dockers y docker-compose.

Docker

En primer lugar se debe realizar la instalación de Docker en la máquina virtual. Para ello [4][11]:

1. Se actualiza la lista de paquetes con el comando `sudo apt update`
2. Se instalan los paquetes necesarios para poder instalar Docker con el comando:

```
sudo apt install apt-transport-https \
ca-certificates curl gnupg-agent \
software-properties-common
```

3. Se añade la clave GPG oficial de Docker con el comando:

```
curl -fsSL \
https://download.docker.com/linux/ubuntu/gpg \
| sudo apt-key add -
```

4. Se añade el repositorio de Docker con:

```
sudo add-apt-repository "deb [arch=amd64] \
https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable"
```

5. Se actualiza con `sudo apt update`
6. Finalmente se instala con:

```
sudo apt install docker-ce-cli:amd64 \
containerd.io
```

7. Para agilizar el uso de comandos en un futuro añadimos el usuario al grupo de Dockers: `sudo usermod -aG docker $USER`
8. Se debe cerrar la terminal y abrir una nueva sesión para que se apliquen los cambios.
9. Para comprobar que todo ha funcionado correctamente, se ejecuta:

```
docker run hello-world
```

docker-compose

Docker Compose es una herramienta para definir y ejecutar aplicaciones Docker de múltiples contenedores. Con Docker Compose, se hace uso de un archivo YAML para configurar los servicios de la aplicación. Luego, con un solo comando, se crean e inician todos los servicios desde la configuración. Para más información, consultar la documentación oficial [3].

1. Descargar la última versión disponible en Github. En la creación de este documento se ha utilizado el siguiente comando:

```
sudo curl -L \
"https://github.com/docker/compose/releases\
/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" \
-o /usr/local/bin/docker-compose
```

2. Cambiar los permisos para ser ejecutable:

```
sudo chmod +x /usr/local/bin/docker-compose
```

3. Comprobación de que está instalado con `docker-compose --version`

A.1.3. Configuración de la aplicación

Se comienza copiando el código del repositorio en la máquina virtual, para esto se arranca una terminal nueva desde la interfaz de Visual Studio y se clona el repositorio mediante SSH o HTTPS. Esto generará una carpeta nueva llamada automl-app con todo el código necesario.

En el fichero llamado `.env.example` se debe configurar con los secretos para el servidor. Para ello, estando dentro de la carpeta automl-app basta con hacer `cp .env.example .env` y modificar el nuevo fichero `.env`. Los parámetros a configurar son:

- MONGODB_ROOT_USER: Nombre de usuario a utilizar en MongoDB
- MONGODB_ROOT_PASSWORD: Contraseña para el usuario definido en MongoDB
- MONGOEXPRESS_USERNAME: Usuario para acceder a Mongo-Express
- MONGOEXPRESS_PASSWORD: Contraseña del usuario
- SERVER_NAME: Nombre de dominio que hemos obtenido en `https://no-ip.com`

- ADMIN_EMAIL: Correo electrónico del administrador para el registro del dominio en *Let's Encrypt*

Con este paso, el código está preparado para ser inicializado, pero aún es necesario configurar el servidor de Nginx.

A.1.4. Generación de certificados HTTPS y configuración de Nginx

Para la generación de los certificados y la configuración de Nginx se ha seguido el tutorial en [14]. El fichero `docker-compose.yaml` ya tiene la configuración básica para poder generar los certificados.

Se ejecuta `init-letsencrypt.sh`, este script realizará la generación de los certificados con la entidad certificadora *Let's Encrypt*. Aunque se ha comprobado que está configuración no debería dar fallos, se recomienda modificar la variable del script `staging` para evitar llegar al límite de fallos en caso de que tengamos alguno en la configuración. Una vez se complete, sin errores, se vuelve a cambiar y se ejecuta de nuevo para obtener los certificados.

Al finalizar, la aplicación ya tendrá los certificados validados que permitirán que aparezca como una página con conexión segura (HTTPS).

A.1.5. Arranque de aplicación

Por último, una vez se han configurado todos los pasos anteriores, se puede arrancar la aplicación web de AutoML. Para ello, es tan sencillo como encontrarse en la carpeta automl-app y ejecutar el comando `docker-compose up -d --build`

Este arrancará los contenedores de Docker con la configuración declarada en el fichero `docker-compose.yaml`. Una vez los servicios estén activos se puede acceder a la página web en el dominio que hayamos obtenido en `https://no-ip.com` desde cualquier navegador.

Apéndice B

Información detallada de los endpoints

A modo de recopilación y, en caso de no tener acceso a la documentación interactiva, se hace una recopilación de los endpoints desarrollados.

B.1. FastAPI

B.1.1. Endpoints de Dataset

- /datasets/upload → Subida de Dataset a MongoDB.
- /datasets/{dataset_id} → Información del Dataset con id *dataset_id*.
- /datasets/{dataset_id}/download → Descarga del Dataset.
- /datasets → Información general de todos los datasets asociados al usuario.
- /datasets/{dataset_id}/dataviz → Generar o recuperar la visualización de datos.
- /datasets/{dataset_id}/modify_dtypes → Modificar los tipos de datos inferidos.

B.1.2. Endpoints de AutoML

- /experiments → Información general de los experimentos del usuario.
- /experiments/upload → Creación de un nuevo experimento.

- /experiments/valid_params → Devuelve los parámetros válidos para la configuración de *OrangePrint*.
- /experiments/{experiment_id} → Información sobre el experimento con id *experiment_id*
- /experiments/{experiment_id}/train → Comenzar el entrenamiento del experimento.
- /experiments/{experiment_id}/results/{metric} → Obtención del modelo con mejor resultado en ciertas métrica.
- /experiments/{experiment_id}/{pipeline_id} → Obtención del modelo por *pipeline_id*
- /experiments/{experiment_id}/{pipeline_id}/predict → Realizar predicción con dicho modelo.

B.1.3. Enpoints de gestión de usuarios

- /register → Registro de usuario en la aplicación.
- /login → Inicio de sesión.
- /logout → Cerrar sesión.

B.1.4. Endpoints comunes

- /check_token → Comprueba la validez de la sesión.
- /check_admin → Comprueba si el usuario es de tipo administrador.

B.2. Flask

Por ser FastAPI un intermediario entre Flask y el módulo de Python AutoML la mayoría de endpoints tienen nombres y funcionalidades similares en Flask.

En flask los valores que sirven cómo parámetros se encapsulan entre <> en lugar de las llaves {} de FastAPI. Por otro lado, la mayor parte de endpoints en Flask cumplen la función de servir un fichero HTML, la ejecución de tareas se realiza en FastAPI.

B.2.1. Endpoints de Dataset

- /datasets/<dataset_id>→ Sirve el HTML con la información del Dataset.
- /datasets/<dataset_id>/report → Muestra el análisis exploratorio del Dataset.
- /datasets/<dataset_id>/delete → Permite eliminar el Dataset.
- /datasets/<dataset_id>/download → Descargar el Dataset.
- /datasets/<dataset_id>/modify_dtypes → Modificar los tipos de datos inferidos.

B.2.2. Endpoints de AutoML

- /experiments/upload → Creación del experimento,
- /experiments/<experiment_id>→ HTML con la información del experimento.
- /experiments/<experiment_id>/delete → Permite borrar el experimento.
- /experiments/<experiment_id>/train → Entrenamiento del experimento.
- /experiments/<experiment_id>/results/<metric>→ Muestra el modelo con mejor resultado en cierta métrica.
- /experiments/<experiment_id>/<pipeline_id>→ Muestra el modelo con id *pipeline_id*.
- /experiments/<experiment_id>/<pipeline_id>/predict → Permite realizar predicciones con el modelo.

B.2.3. Endpoints de gestión de usuario

- / → Sirve la página de presentación de la aplicación.
- /register → Página de registro de usuario.
- /login → Inicio de sesión
- /logout → Cierre de sesión.

B.2.4. Endpoints del centro de mandos (Dashboard)

- /dashboard → Muestra la web del centro de mandos.

B.2.5. Endpoints de administrador

- /admin → Panel de administrador.
- /admin/mongo-express → Interfaz de *Mongo-Express*
- /admin/api/docs → Documentación interactiva de FastAPI

Bibliografía

[1] Moez Ali. Pycaret: An open source, low-code machine learning library in python. https://www.pycaret.org, April 2020. PyCaret version 1.0.0.

[2] AWS Marketplace. AWS Marketplace - DataRobot Pricing. https://aws.amazon.com/marketplace/pp/prodview-qmlu2m6haaf4i. Accedido en Mayo 22, 2023.

[3] Docker Contributors. Docker compose overview. https://docs.docker.com/compose/.

[4] Docker Contributors. Install docker engine on ubuntu. https://docs.docker.com/engine/install/ubuntu/.

[5] MDN Contributors. Glosario de mdn web docs: Definiciones de términos relacionados con la web - crud. https://developer.mozilla.org/es/docs/Glossary/CRUD.

[6] Radwa Elshawi, Mohamed Maher, and Sherif Sakr. Automated machine learning: State-of-the-art and open challenges, 2019.

[7] Hugo Jair Escalante. Automated machine learning – a brief review at the end of the early years, 2020.

[8] Google Cloud Platform. DataRobot AI Cloud Platform for Google Cloud. https://console.cloud.google.com/marketplace/product/datarobot-public/datarobot-ai-cloud-platform-for-google-cloud. Accedido en Junio 19, 2023.

[9] Google Cloud Platform. Pricing MV Google. https://cloud.google.com/products/calculator#id=b6c289d1-5abd-4e22-8512-1fd0cf97b34b. Accedido en Marzo 22, 2023.

[10] Xin He, Kaiyong Zhao, and Xiaowen Chu. AutoML: A survey of the state-of-the-art. *Knowledge-Based Systems*, 212:106622, jan 2021.

[11] Brian Hogan. How to install and use docker on ubuntu 20.04. https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-20-04.

[12] IBM Corporation. IBM List Prices for Driveless AI. https://www.ibm.com/downloads/cas/US-ENUS219-434-CA/name/ENUS-219-434-LIST_PRICES_2019_10_22.PDF, 2019. Accedido en Mayo 22, 2023.

[13] OWASP. Owasp top ten project. https://owasp.org/www-project-top-ten/, 2023. Accedido en Mayo 20, 2023.

[14] Philipp. Nginx and let's encrypt with docker in less than 5 minutes. https://pentacent.medium.com/nginx-and-lets-encrypt-with-docker-in-less-than-5-minutes-b4b8a60d3a71, 2018. Accedido en Febrero 12, 2023.

[15] Defuse Security. Salted password hashing - doing it right. https://crackstation.net/hashing-security.htm#properhashing.