UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

Digitalización del Boletín de Turismo del Observatorio de la Provincia de Burgos Documentación Técnica

Presentado por Christian Andrés Núñez Duque
en Universidad de Burgos — 7 de julio de 2025
Tutor: Bruno Baruque Zanón
Cotutor: Julio César Puche Regaliza

Índice general

Índice de figuras

Índice de tablas

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En este apéndice se pretende detallar el plan de proyecto software seguido para el desarrollo del mismo. Para ello, se mostrará la planificación temporal, el estudio de viabilidad que comprende aspectos relevantes de su viabilidad económica y legal.

A.2. Planificación temporal

El proyecto se ha desarrollado siguiendo el modelo de desarrollo ágil SCRUM. Para ello, se han definido una serie de sprints de 2 semanas utilizando GitHub Projects. En cada sprint se ha tratado de implementar una serie de funcionalidades definidas previamente como issues.

Sprint 1 (27/02/2025 - 12/03/2025)

Este primer sprint se centró en la creación del proyecto y en su inicio. Dado que este proyecto se basa en un TFG anterior, algunas tareas están relacionadas con ellas.

- Leer y analizar el TFG previo relacionado con el tema.
- Definir el enfoque y tecnologías a utilizar en el proyecto.
- Probar diferentes APIs (de extracción de datos).
- Probar funcionalidades de Power BI.

- Probar extracciones de reviews a partir de GeoJSON de OpenStreetMaps.
- Probar funcionalidades de PowerPages.
- Probar la viabilidad de extraer datos con Apify.
- Probar Google Maps Scraper.

Sprint 2 (13/03/2025 - 26/03/2025)

En el segundo sprint, se trató de iniciar el proceso de documentación del proyecto y de iniciar el sistema de extracción de datos.

- Obtener place ids sin usar los cuadrantes de las coordenadas.
- Iniciar el proceso de documentación
- Obtener place ids a partir de textSearch.
- Documentar los objetivos del proyecto.
- Actualizar documentación de los objetivos del proyecto.
- Automatizar la extracción de los datos del INE a través de su API. (Cerrada como no planeada)

Sprint 3 (27/03/2025 - 09/04/2025)

Para el tercer sprint, el objetivo principal era diseñar el modelo de datos, el cual fue actualizado en los últimos sprints.

- Diseñar un esquema para el modelo de datos.
- Añadir manualmente los datos (poblacion y coordenadas) de los municipios que no provee el INE. (Cerrada como no planeada)

Sprint 4 (10/04/2025 - 23/04/2025)

En el cuarto sprint, se inició a desarrollar el sistema de aprendizaje automático y el primer modelo de red neuronal.

- Crear una red neuronal a partir de reviews.

Sprint 5 (24/04/2025 - 07/05/2025)

En el quinto sprint se trató de crear el modelo final de extracción de datos.

- Probar la extracción de reviews mediante Overpass QL, Google Places API y Apify.
- Crear una base de datos para almacenar las reviews.

Sprint 6 (08/05/2025 - 21/05/2025)

Durante el sexto sprint, tras no haber obtenido buenos resultados con el modelo de red neuronal, se optó por construir un dataset propio para el modelo de aprendizaje automático.

- Crear un dataset con reviews y categorías de los POIs.

Sprint 7 (22/05/2025 - 04/06/2025)

El séptimo sprint se basó en la creación de un diccionario para mapear subcategorías a categorías para mejorar el modelo de aprendizaje automático. Finalmente esto fue descartado tras rediseñar el conjunto de datos.

- Hacer un mapeo de categorías a subcategorías (Cerrada como no planeada).

Sprint 8 (05/06/2025 - 18/06/2025)

El octavo sprint se centró en redactar la documentación de la memoria del proyecto. El objetivo era crear una versión inicial pero completa de la memoria.

- Añadir conceptos teóricos a la documentación.
- Añadir técnicas y herramientas a la documentación.
- Añadir aspectos relevantes a la documentación.
- Añadir los trabajos relacionados a la documentación.

Sprint 9 (19/06/2025 - 02/07/2025)

El penúltimo sprint se centró en completar la memoria y crear el cuadro de mando.

- Añadir la introducción a la documentación.
- Añadir las conclusiones y líneas de trabajo futuras a la documentación.
- Crear un dashboard de Power BI para la visualización de resultados.
- Crear una Azure Function para la predicción y guardados automáticos sobre los recursos.

Sprint 10 (03/07/2025 - 07/07/2025)

En el último sprint se ha tratado de completar toda la documentación y el cuadro de mando. Además se ha ordenado el repositorio y se han actualizado los ficheros antiguos.

- Implementar un sistema de análisis y detección de defectos de código.
- Crear un sistema de recomendación a partir de la localización y reseñas de los recursos.
- Añadir el plan de proyecto a los anexos.
- Añadir los requisitos a los anexos.
- Añadir la especificación de diseño a los anexos.
- Añadir el manual de programador a los anexos.
- Añadir el manual de usuario a los anexos.
- Añadir el anexo de sostenibilización curricular.

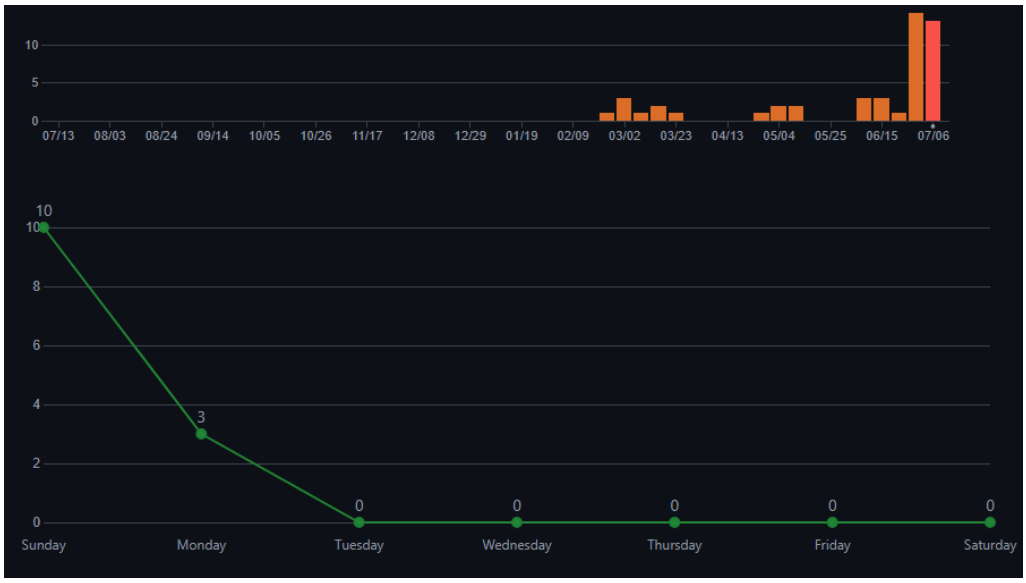

Figura A.1: Gráfica de commits

A.3. Estudio de viabilidad

Viabilidad económica

Desde el punto de vista económico, hay varios aspectos a tener en cuenta:

Costes de hardware

Este proyecto ha sido desarrollado en un ordenador de sobremesa con un coste aproximado de 800 euros comprado hace 9 años por lo que se puede considerar amortizado.

Extracción de datos

Para la extracción de datos se ha optado por utilizar la plataforma Apify que permite extraer datos de forma masiva de páginas web y APIs. Se muestra una tabla con los precios de las cuentas de Apify[5] en dólares américanos:

Tipo de cuenta	Número de reviews	Precio/mes al 50 %
Free	14285	0$
Starter	111428	19,5$
Scale	568571	99,5$
Business	2854285	499,5$
Enterprise	Ilimitado	Hablar con Apify

Tabla A.1: Costos por solicitud en Apify

Cabe destacar que ofrecen un descuento del 50 % para cuentas educativas.

Se estiman alrededor de un millón y medio de reseñas en toda la provincia de Burgos, por lo que la mejor opción puede parecer la cuenta Business. Sin embargo, hay muchos recursos con pocas reseñas o sin reseñas que podrían ser descartados por lo que la mejor opción serían 2 cuentas Scale.

Estos pagos son mensuales, por lo que se estima un coste de 198$ al mes en caso de que cada mes se quiera realizar una nueva extracción. Haciendo la conversión a euros, a día 3 de julio de 2025, el coste sería de **167,92 euros al mes**. (1 USD = 0,85 EUR)

Si se decide pagar anualmente, se aplicaría un descuento del 10 % sobre cada cuenta, lo que supondría un coste de 179$ al mes entre las dos cuentas y de 2148$ anualmente. Haciendo la conversión a euros, el coste sería de **1821,63 euros al año**. (1 USD = 0,85 EUR)

Estos costes han sido calculados teniendo en cuenta que se aplicaría un descuento del 50 % por cuentas educativas. Cabe destacar que estos cálculos están pensados para una futura escalabilidad del proyecto. Para el desarrollo del mismo se ha optado por utilizar cuentas gratuitas y un scraper de código abierto gratuito.

Costes de alojamiento

Para el alojamiento de la base de datos, se ha optado por utilizar un servidor en la nube de Microsoft Azure. La opción elegida ha sido la opción serverless ya que es la más económica y permite alojar la base de datos correctamente. En el modelo Serverless, se paga 5,69 USD al mes por 41,6 GB de almacenamiento, además de un coste adicional de 0,000159 USD por cada segundo de procesamiento de la CPU. En cambio, con un servidor dedicado, el coste mensual asciende a 396,10 USD por los mismos 41,6 GB de almacenamiento, incluyendo dos núcleos de CPU. Respecto al alojamiento del cuadro de mando en PowerPages es necesaria una cuenta de Power BI Pro

que tiene un coste de 13,10 euros al mes. De esta forma se estima un coste mensual de entre 6 y 8 USD, lo que al cambio a euros sería aproximadamente de **5 a 7 euros al mes**. (1 USD = 0,85 EUR)

Viabilidad legal

Protección de datos

Respecto a la protección de datos, se ha procurado mantener en el anónimato los datos extraídos. No se trata de un proyecto con ánimo de lucro por lo que no se pretende obtener ningún beneficio económico de los datos extraídos. En ningún momento del cuadro de mando se muestra información personal de los usuarios, únicamente se muestra la distinción entre género.

Extracción de datos

Si bien es cierto que la extracción de datos mediante scraper es un tema controvertido, en España, realizar web scraping es completamente legal. En este proyecto, la información extraída es de acceso público y no se está vulnerando ningún derecho de propiedad intelectual.[3]

Licencias de software

Durante el desarrollo de este proyecto se han utilizado las siguientes herramientas con sus correspondientes licencias software:

Herramienta	**Versión**	**Licencia**
Visual Studio Code	1.101.2	MIT
LaTeX	2023	LPPL
Google Colab	N/A	Propietaria
Git	2.47.1	GPLv2
GitHub	N/A	Propietaria
Python	3.10.11	PSF
TensorFlow	2.18.0	Apache 2.0
Keras	3.8.0	Apache 2.0
Numpy	2.0.2	BSD
Scikit Learn	1.6.1	BSD
Pysentimiento	0.7.3	MIT
Emoji	2.14.1	BSD
Pyodbc	5.2.0	MIT
Langdetect	1.0.9	Apache 2.0
Gender_guesser	0.4.0	GPLv3
Deep_translator	1.11.4	MIT
Matplotlib	3.10.0	PSF
Seaborn	0.13.2	BSD
Pandas	2.2.2	GPLv3
Transformers	4.53.0	Apache 2.0
Apify_client	1.10.0	Apache 2.0
Requests	2.32.4	Apache 2.0
SSMS	21.2.5	Propietaria
PowerBI	2.140.1577.0	Propietaria
Draw.io	27.0.6	Apache 2.0
Docker	27.4.0	Apache 2.0
SonarQube	24.0.2	GPLv3
Google Maps Scraper	1.0	MIT
8icons	2025	Icons8 UML
httpx	0.28.1	BSD
Dotenv	1.1.1	BSD

Tabla A.2: Tabla de herramientas y licencias

Propiedad intelectual

El proyecto se puede dividir en varios productos finales con diferentes licencias:

Recurso	**Licencia**
Dataset	CC BY-NC-SA
Cuadro de mando	CC BY-NC-SA
Código fuente	CC BY-NC-SA
Documentación	CC BY-NC-SA

Tabla A.3: Tabla de productos y licencias

Esto implica que el dataset, cuadro de mando y documentación pueden ser utilizados por cualquier persona siempre que se mantenga la misma licencia y se reconozca al autor original, pero no se permite su uso comercial. El código fuente, sin embargo, se puede utilizar, modificar y distribuir libremente siempre que se mantenga la misma licencia y se reconozca al autor original.

Apéndice B

Especificación de Requisitos

B.1. Introducción

En este anexo se especifican los requisitos funcionales y no funcionales del sistema, así como los casos de uso que describen las interacciones entre los actores y el sistema. A continuación se listan los objetivos de los que nacen los requisitos funcionales y no funcionales junto a los casos de uso que los implementan.

B.2. Objetivos generales

Este proyecto cuenta con varios objetivos generales detallados a continuación:

- Automatizar la extracción de recursos y reseñas de Google Maps.
- Desarrollar una red neuronal que permita clasificar los recursos según su tipo.
- Analizar los datos obtenidos con herramientas de inteligencia de negocio para obtener métricas y visualizaciones útiles.
- Publicar los resultados en un sitio web para que sean accesibles al público.

B.3. Catálogo de requisitos

Requisitos funcionales

- **RF-1 Extraer los datos:** El sistema debe permitir la extracción de datos de Google Maps.
  - **RF-1.1 Extraer recursos:** El sistema debe extraer información de recursos como restaurantes, hoteles, etc.
  - **RF-1.2 Extraer reseñas:** El sistema debe extraer reseñas de los recursos obtenidos.
  - **RF-1.3 Configurar parámetros de extracción:** El sistema debe permitir la configuración de parámetros como número de reseñas, idioma, etc.
- **RF-2 Clasificar con aprendizaje automático:** El sistema debe clasificar los recursos extraídos utilizando una red neuronal.
  - **RF-2.1 Entrenar la red neuronal:** El sistema debe permitir el entrenamiento de la red neuronal con los datos extraídos.
  - **RF-2.2 Clasificar recursos:** El sistema debe clasificar los recursos en diferentes categorías (por ejemplo, restaurantes, hoteles, etc.).
  - **RF-2.3 Evaluar la precisión del modelo:** El sistema debe evaluar la precisión del modelo de clasificación utilizando métricas adecuadas.
  - **RF-2.4 Ajustar el modelo:** El sistema debe permitir ajustar el modelo de clasificación para mejorar su precisión.
  - **RF-2.5 Guardar el modelo:** El sistema debe guardar el modelo entrenado para su uso posterior.
  - **RF-2.6 Cargar el modelo:** El sistema debe permitir cargar un modelo previamente entrenado para su uso.
  - **RF-2.7 Predecir tipos de recursos:** El sistema debe permitir predecir los tipos de nuevos recursos utilizando el modelo entrenado.
- **RF-3 Analizar los resultados:** El sistema debe generar informes y visualizaciones a partir de los datos analizados.
  - **RF-3.1 Visualizar el TORI:** El sistema debe permitir visualizar el TORI de los recursos.

  - **RF-3.2 Visualizar el número de reseñas:** El sistema debe permitir visualizar el número de reseñas por recurso.
  - **RF-3.3 Visualizar la puntuación media:** El sistema debe permitir visualizar la puntuación media de los recursos.
  - **RF-3.4 Visualizar la distribución de categorías:** El sistema debe permitir visualizar la distribución de recursos por categoría.
  - **RF-3.5 Visualizar la evolución temporal:** El sistema debe permitir visualizar la evolución de los recursos a lo largo del tiempo.
  - **RF-3.6 Visualizar el mapa de recursos:** El sistema debe permitir visualizar un mapa con la ubicación de los recursos.
- **RF-4 Publicar los resultados:** El sistema debe publicar los resultados en un sitio web accesible al público.

Requisitos no funcionales

- **RNF-1 Disponibilidad:** El sistema debe estar disponible 24/7 para la extracción y análisis de datos.
- **RNF-2 Rendimiento:** El sistema debe ser capaz de procesar grandes volúmenes de datos en un tiempo razonable.
- **RNF-3 Usabilidad:** El sistema debe ser fácil de usar para los usuarios finales.
- **RNF-4 Seguridad:** El sistema debe garantizar la seguridad de los datos extraídos y analizados.
- **RNF-5 Portabilidad:** El sistema debe ser portablec por tanto, debe poder ejecutarse en diferentes entornos y máquinas.

B.4. Especificación de requisitos

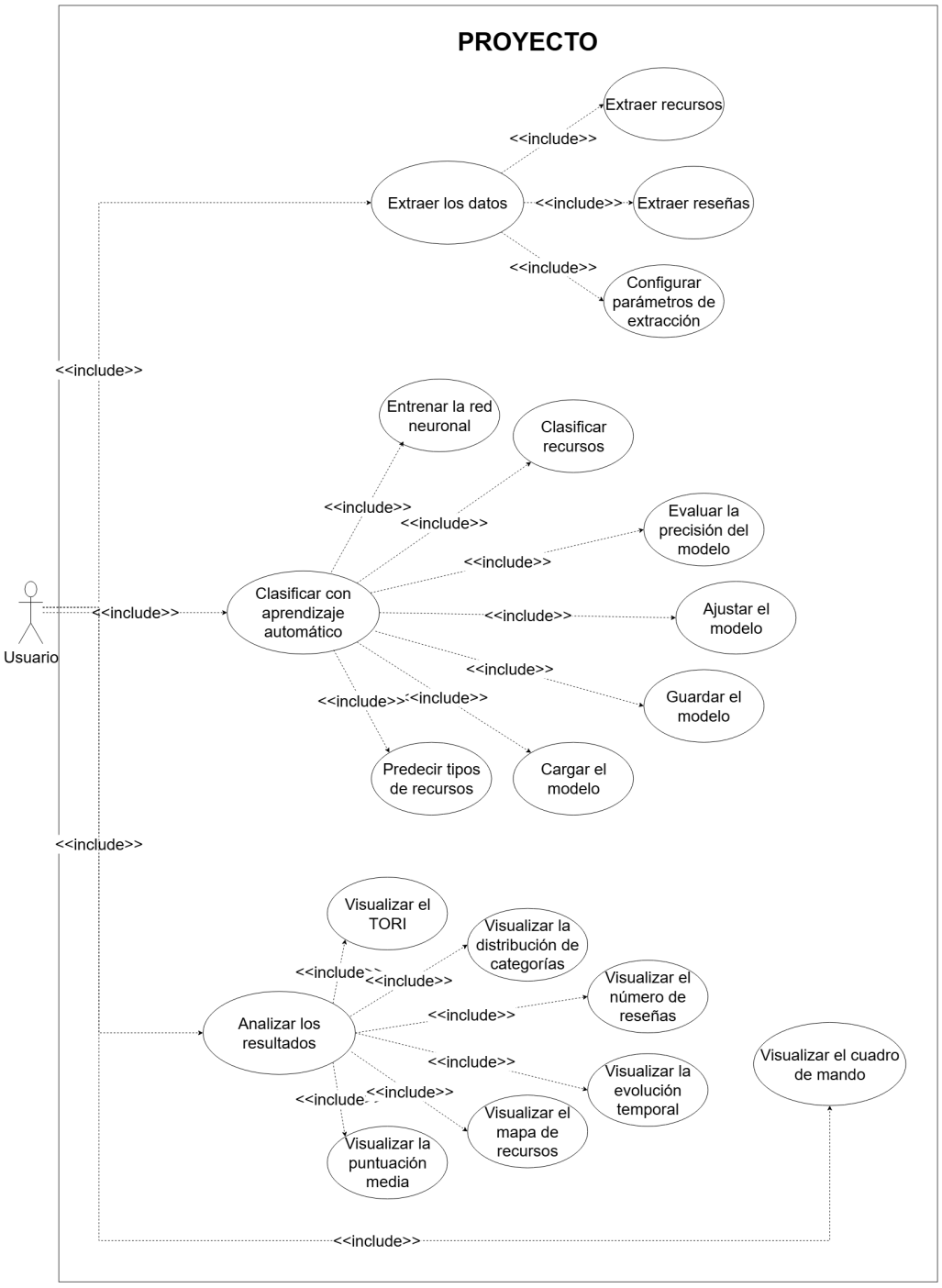

Figura B.1: Diagrama de casos de uso

CU-1	**Extraer los datos**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-1, RF-1.1, RF-1.2, RF-1.3
Descripción	Se deben extraer las reseñas y recursos y poder configurar su extracción
Precondición	Poseer una API KEY de Apify
Acciones	1. Determinar la API KEY de Apify en el fichero .env 2. Seleccionar los recursos a extraer mediante placeID 3. Seleccionar los parámetros de extracción (número de reseñas, idioma, etc.) 4. Ejecutar el script de extracción
Postcondición	Fichero .csv con los datos extraídos
Excepciones	Caída de Apify
Importancia	Alta

Tabla B.1: CU-1 Extraer los datos

CU-2	**Extraer los recursos**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-1.1
Descripción	Se deben extraer los recursos de forma sencilla
Precondición	Poseer una API KEY de Apify
Acciones	1. Determinar la API KEY de Apify en el fichero .env 2. Seleccionar los recursos a extraer mediante placeID 3. Ejecutar el script de extracción
Postcondición	Fichero .csv con los recursos extraídos
Excepciones	Caída de Apify
Importancia	Alta

Tabla B.2: CU-2 Extraer los recursos

CU-3	**Extraer las reseñas**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-1.2
Descripción	Se deben extraer las reseñas de forma sencilla
Precondición	Poseer una API KEY de Apify
Acciones	1. Determinar la API KEY de Apify en el fichero .env 2. Seleccionar los recursos a extraer mediante placeID 3. Seleccionar los parámetros de extracción (número de reseñas, idioma, etc.) 4. Ejecutar el script de extracción
Postcondición	Fichero .csv con las reseñas extraídas
Excepciones	Caída de Apify
Importancia	Alta

Tabla B.3: CU-3 Extraer las reseñas

CU-4	**Configurar parámetros de extracción**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-1.3
Descripción	El usuario debe poder configurar cómodamente los parámetros de la extracción de datos
Precondición	Poseer una API KEY de Apify
Acciones	1. Determinar la API KEY de Apify en el fichero .env 2. Seleccionar los recursos a extraer mediante placeID 3. Seleccionar los parámetros de extracción (número de reseñas, idioma, etc.) 4. Ejecutar el script de extracción
Postcondición	Fichero .csv con los datos extraídos según los parámetros seleccionados
Excepciones	Caída de Apify
Importancia	Alta

Tabla B.4: CU-4 Configurar parámetros de extracción

CU-5	**Clasificar con aprendizaje automático**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-2, RF-2.1, RF-2.2, RF-2.3, RF-2.4, RF-2.5, RF-2.6, RF-2.7
Descripción	El usuario debe poder utilizar la red neuronal para predecir tipos de recursos
Precondición	Poseer el script de la red neuronal y el conjunto de datos
Acciones	1. Cargar el script/notebook en Google Colab 2. Seleccionar GPU como entorno de ejecución 3. Cargar el conjunto de datos de entrenamiento 4. Ejecutar el script/notebook para entrenar la red neuronal 5. Cargar el conjunto de recursos a clasificar 6. Ejecutar el script/notebook para clasificar los recursos
Postcondición	Fichero .sql con los recursos clasificados
Excepciones	Caída de Google Colab Caída de Azure
Importancia	Alta

Tabla B.5: CU-5 Clasificar con aprendizaje automático

CU-6	**Entrenar la red neuronal**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-2.1
Descripción	El usuario debe poder reentrenar la red neuronal en cualquier momento
Precondición	Poseer el script de la red neuronal y el conjunto de datos
Acciones	1. Cargar el script/notebook en Google Colab 2. Seleccionar GPU como entorno de ejecución 3. Cargar el conjunto de datos de entrenamiento 4. Ejecutar el script/notebook para entrenar la red neuronal
Postcondición	Ficheros .keras y .pkl con el modelo entrenado
Excepciones	Caída de Google Colab Caída de Azure
Importancia	Media

Tabla B.6: CU-6 Entrenar la red neuronal

CU-7	**Clasificar recursos**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-2.2
Descripción	El usuario debe poder clasificar recursos a partir del modelo
Precondición	Poseer los ficheros del modelo y de los recursos a clasificar
Acciones	1. Cargar el script/notebook en Google Colab 2. Seleccionar GPU como entorno de ejecución 3. Cargar los ficheros del modelo (si no están ya cargados) y los recursos a clasificar 4. Ejecutar el script/notebook para clasificar recursos
Postcondición	Fichero .sql con los recursos clasificados
Excepciones	Caída de Google Colab Caída de Azure
Importancia	Alta

Tabla B.7: CU-7 Clasificar recursos

CU-8	**Evaluar la precisión del modelo**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-2.3
Descripción	El usuario debe poder ver el informe de clasificación del modelo y la matriz de confusión
Precondición	Poseer el script de la red neuronal y el conjunto de datos
Acciones	1. Cargar el script/notebook en Google Colab 2. Seleccionar GPU como entorno de ejecución 3. Cargar el conjunto de datos de entrenamiento 4. Ejecutar el script/notebook para entrenar la red neuronal
Postcondición	Informe de clasificación y matriz de confusión en la terminal
Excepciones	Caída de Google Colab Caída de Azure
Importancia	Media

Tabla B.8: CU-8 Evaluar la precisión del modelo

CU-9	**Ajustar el modelo**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-2.4
Descripción	El usuario debe poder ajustar los hiperparámetros del modelo
Precondición	Poseer el script de la red neuronal y el conjunto de datos
Acciones	1. Cargar el script/notebook en Google Colab 2. Seleccionar GPU como entorno de ejecución 3. Cargar el conjunto de datos de entrenamiento 4. Ajustar los hiperparámetros del modelo (número de capas, neuronas, tasa de aprendizaje, etc.) 5. Ejecutar el script/notebook para entrenar la red neuronal
Postcondición	Informe de clasificación y matriz de confusión en la terminal
Excepciones	Caída de Google Colab Caída de Azure
Importancia	Media

Tabla B.9: CU-9 Evaluar la precisión del modelo

CU-10	**Guardar el modelo**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-2.5
Descripción	El usuario debe poder guardar el modelo para su posterior uso
Precondición	Poseer el script de la red neuronal y el conjunto de datos
Acciones	1. Cargar el script/notebook en Google Colab 2. Seleccionar GPU como entorno de ejecución 3. Cargar el conjunto de datos de entrenamiento 4. Ejecutar el script/notebook para entrenar la red neuronal
Postcondición	Ficheros .keras y .pkl con el modelo entrenado
Excepciones	Caída de Google Colab Caída de Azure
Importancia	Baja

Tabla B.10: CU-10 Guardar el modelo

CU-11	**Cargar el modelo**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-2.6
Descripción	El usuario debe poder cargar el modelo para la predicción de tipos de recurso
Precondición	Poseer los ficheros del modelo
Acciones	1. Cargar el script/notebook en Google Colab 2. Seleccionar GPU como entorno de ejecución 3. Cargar los ficheros del modelo (si no están ya cargados) y los recursos a clasificar
Postcondición	Mensaje en la terminal de "modelo cargado"
Excepciones	Caída de Google Colab Caída de Azure
Importancia	Baja

Tabla B.11: CU-11 Cargar el modelo

CU-12	**Predecir tipos de recurso**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-2.7
Descripción	El usuario debe poder predecir tipos de recursos
Precondición	Poseer los ficheros del modelo
Acciones	1. Cargar el script/notebook en Google Colab 2. Seleccionar GPU como entorno de ejecución 3. Cargar los ficheros del modelo (si no están ya cargados) y los recursos a clasificar 4. Ejecutar el script/notebook para clasificar recursos
Postcondición	Fichero .sql con los recursos clasificados
Excepciones	Caída de Google Colab Caída de Azure
Importancia	Alta

Tabla B.12: CU-12 Predecir tipos de recurso

CU-13	**Analizar los resultados**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-3, RF-3.1, RF-3.2, RF-3.3, RF-3.4, RF-3.5, RF-3.6
Descripción	El usuario debe poder visualizar el análisis de resultados
Precondición	Poseer el fichero .pbix de PowerBI con el cuadro de mando y las credenciales de acceso a la base de datos
Acciones	1. Ejecutar el fichero .pbix de PowerBI 2. Escribir las credenciales de acceso a la base de datos 3. Visualizar el cuadro de mando con los datos analizados
Postcondición	Ver métricas y gráficas
Excepciones	Caída de Azure
Importancia	Alta

Tabla B.13: CU-13 Analizar los resultados

CU-14	**Visualizar el TORI**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-3.1
Descripción	El usuario debe poder visualizar el TORI
Precondición	Poseer el fichero .pbix de PowerBI con el cuadro de mando y las credenciales de acceso a la base de datos
Acciones	1. Ejecutar el fichero .pbix de PowerBI 2. Escribir las credenciales de acceso a la base de datos 3. Visualizar el TORI de los recursos
Postcondición	Ver el TORI de los recursos
Excepciones	Caída de Azure
Importancia	Alta

Tabla B.14: CU-14 Visualizar el TORI

CU-15	**Visualizar el número de reseñas**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-3.2
Descripción	El usuario debe poder visualizar el número de reseñas
Precondición	Poseer el fichero .pbix de PowerBI con el cuadro de mando y las credenciales de acceso a la base de datos
Acciones	1. Ejecutar el fichero .pbix de PowerBI 2. Escribir las credenciales de acceso a la base de datos 3. Visualizar el número de reseñas de los recursos
Postcondición	Ver el número de reseñas de los recursos
Excepciones	Caída de Azure
Importancia	Media

Tabla B.15: CU-15 Visualizar el número de reseñas

CU-16	**Visualizar la puntuación media**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-3.3
Descripción	El usuario debe poder visualizar la puntuación media de los recursos
Precondición	Poseer el fichero .pbix de PowerBI con el cuadro de mando y las credenciales de acceso a la base de datos
Acciones	1. Ejecutar el fichero .pbix de PowerBI 2. Escribir las credenciales de acceso a la base de datos 3. Visualizar la puntuación media de los recursos
Postcondición	Ver la puntuación media de los recursos
Excepciones	Caída de Azure
Importancia	Media

Tabla B.16: CU-16 Visualizar la puntuación media

CU-17	**Visualizar la evolución temporal**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-3.5
Descripción	El usuario debe poder visualizar la evolución temporal de las reseñas
Precondición	Poseer el fichero .pbix de PowerBI con el cuadro de mando y las credenciales de acceso a la base de datos
Acciones	1. Ejecutar el fichero .pbix de PowerBI 2. Escribir las credenciales de acceso a la base de datos 3. Visualizar la evolución temporal de las reseñas
Postcondición	Ver la evolución temporal de las reseñas
Excepciones	Caída de Azure
Importancia	Media

Tabla B.17: CU-17 Visualizar la evolución temporal

CU-18	**Visualizar el mapa de recursos**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-3.6
Descripción	El usuario debe poder visualizar el mapa de recursos
Precondición	Poseer el fichero .pbix de PowerBI con el cuadro de mando y las credenciales de acceso a la base de datos
Acciones	1. Ejecutar el fichero .pbix de PowerBI 2. Escribir las credenciales de acceso a la base de datos 3. Visualizar el mapa de recursos
Postcondición	Ver el mapa de recursos
Excepciones	Caída de Azure
Importancia	Media

Tabla B.18: CU-18 Visualizar el mapa de recursos

CU-19	**Publicar los resultados**
Versión	1.0
Autor	Christian Andrés Núñez Duque
Requisitos asociados	RF-3
Descripción	El usuario debe poder visualizar el cuadro de mando a través de una página web
Precondición	Poseer la URL de la web
Acciones	1. Abrir el navegador web 2. Introducir la URL del cuadro de mando
Postcondición	Ver el cuadro de mando
Excepciones	Caída de Azure Caída de Internet
Importancia	Alta

Tabla B.19: CU-19 Publicar los resultados

Apéndice C

Especificación de diseño

C.1. Introducción

En este apéndice se detallan los aspectos de diseño del sistema, incluyendo el diseño de datos, arquitectónico y procedimental. Esto se acompañará de diagramas que faciliten la comprensión de la estructura y funcionamiento del sistema. También se incluyen capturas de pantalla de la aplicación final para poder visualizar su interfaz y funcionalidades.

C.2. Diseño de datos

En este proyecto, se ha optado por utilizar varias opciones para tratar con los datos dependiendo de la fase del proyecto en la que se trabaje.

Fase de extracción de datos

Vía Apify

Para comenzar esta fase se ha realizado una consulta la API de Overpass para obtener nodos de OSM. Estos datos se recogen en un fichero JSON que posteriormente se procesa para extraer la información relevante. Este fichero JSON está formado por campos y valores. Hay una clave general llamada 'elements' que contiene una lista de nodos como elementos individuales. Cada nodo tiene varios campos, como 'id', 'lat', 'lon', 'tags', etc. Los tags son un diccionario que contiene pares clave-valor, donde las claves son los nombres de los atributos y los valores son sus correspondientes valores.

Una vez obtenido el JSON, se ha pasado por la API de Google para obtener placeIds a partir de las coordenadas obteniendo un fichero CSV con coordenadas y placeIds. Finalmente, se procesa este fichero CSV por un script de Python que realiza llamadas a la API de Apify para recoger las reseñas y recursos en otro fichero CSV. Este fichero está formado por los siguientes campos: 'categoryName', 'city', 'reviewsCount', 'totalScore', 'stars', 'state', 'text', 'title', 'location', 'originalLanguage', 'publishedAtDate', 'placeId', 'url', 'name'.

Vía Google Maps Review Scraper

Al utilizar este scraper, se ha optado por un enfoque diferente. Este scraper devuelve dos ficheros CSV con multitud de campos entre los que se encuentran los recogidos a través de Apify.

Fase de preprocesamiento de datos

El preprocesamiento de datos se divide en tres scripts de Python que se encuentran en el directorio src/dataPreprocessing.

El primer script toma los ficheros CSV obtenidos vía Google Maps Review Scraper y los procesa para obtener un solo fichero CSV similar al obtenido con Apify. Esto implica que este paso no es necesario si se ha optado por el scraper de Apify. Aquí se añade el campo 'originalLanguage', 'city' y se cambia el formato de la fecha de la review.

El segundo script toma el fichero CSV obtenido vía Apify o vía paso anterior y se encarga de detectar el idioma de las reseñas y de traducirlas a español si es que no están en español. Para no perder la reseña original, se añade el campo 'text_original' que contiene la reseña sin traducir.

Finalmente, el tercer script limpia saltos de línea y valores nulos de las reseñas y elimina las que tienen menos de 15 palabras devolviendo un fichero CSV limpio para usar el modelo entrenado.

Fase de predicciones y recomendaciones

Esta fase se encuentra tras el entrenamiento del modelo. En ella se obtienen dos ficheros .sql que contienen los INSERTS que se utilizarán para cargar en la base de datos los recursos con su categoría y los clusters para las recomendaciones.

Modelo de datos

El modelo de datos utilizado para almacenar toda la información utilizada en el cuadro de mando viene dado por el siguiente esquema.

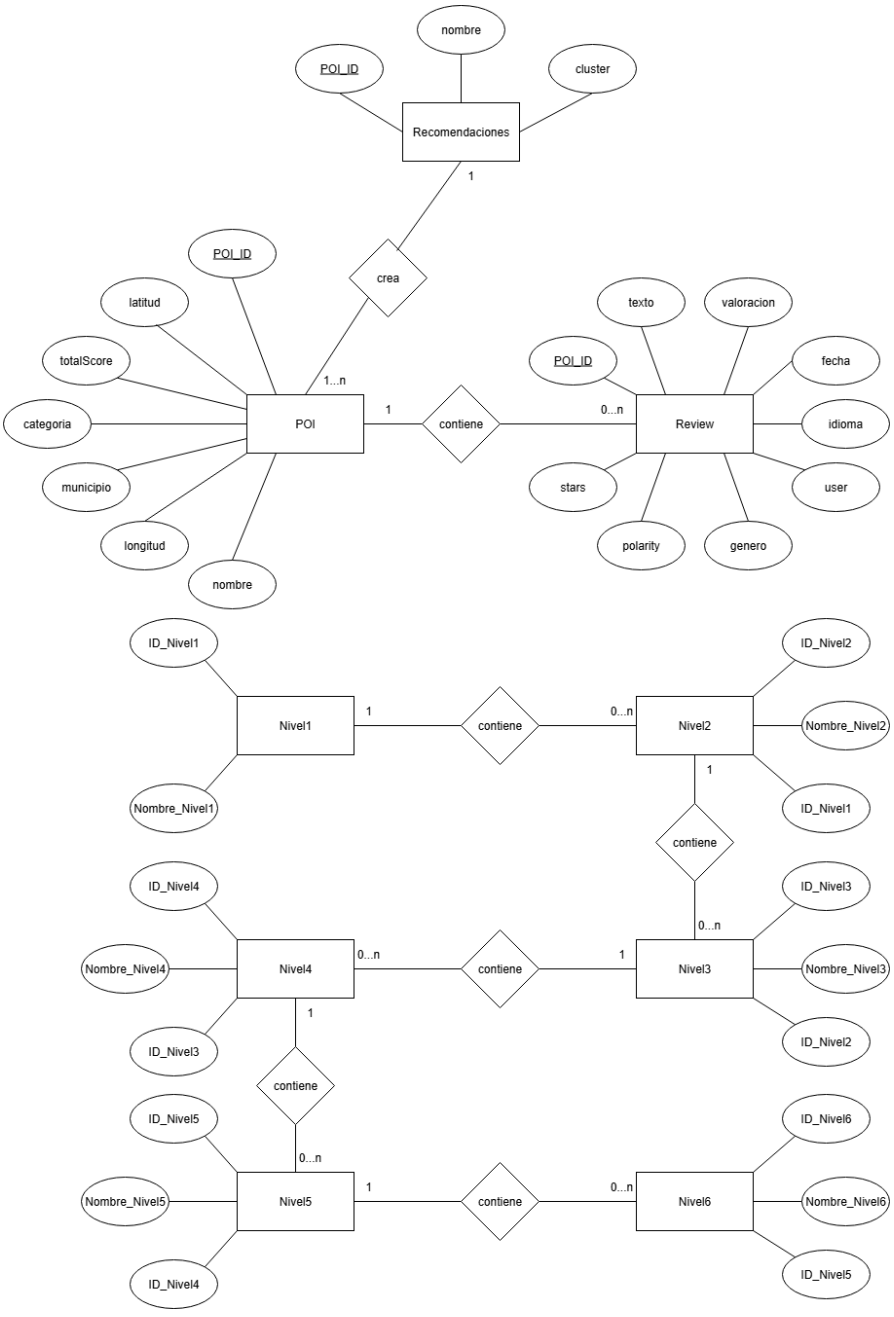

Figura C.1: Diagrama entidad-relación

C.3. Diseño arquitectónico

Este proyecto sigue una arquitectura basada en scripts de Python que se ejecutan secuencialmente como un pipeline. En este enfoque se pueden diferenciar 6 fases del proyecto definidas y diferenciadas claramente por los diferentes scripts:

Extracción de datos

Es la fase inicial del proyecto donde se obtienen los datos de las reseñas y los recursos. Se utilizan dos enfoques diferentes: uno a través de la API de Apify y otro a través del Google Maps Review Scraper. Ambos enfoques generan ficheros CSV con la información necesaria para el análisis posterior.

Preprocesamiento de datos

Es la segunda fase del proyecto donde se procesan los datos obtenidos en la fase anterior. Se realizan varias tareas como la detección del idioma de las reseñas, la traducción al español, la limpieza de saltos de línea y la eliminación de reseñas con menos de 15 palabras. Esta fase es crucial para garantizar que los datos estén en un formato adecuado para el análisis posterior.

Creación del conjunto de datos de entrenamiento

Asociado a la fase de preprocesamiento se encuentra la creación del dataset de entrenamiento. Este dataset ha sido creado por mí de forma manual a partir de las reseñas obtenidas en los pasos anteriores. Para ello, he extraído reseñas de varios recursos seleccionados manualmente a partir de Google Maps y he reetiquetado cada reseña con la categoría del recurso al que pertenece. Este dataset se encuentra en el directorio `src/trainingDataset.csv`.

A continuación, se muestran varias métricas obtenidas al analizar este dataset.

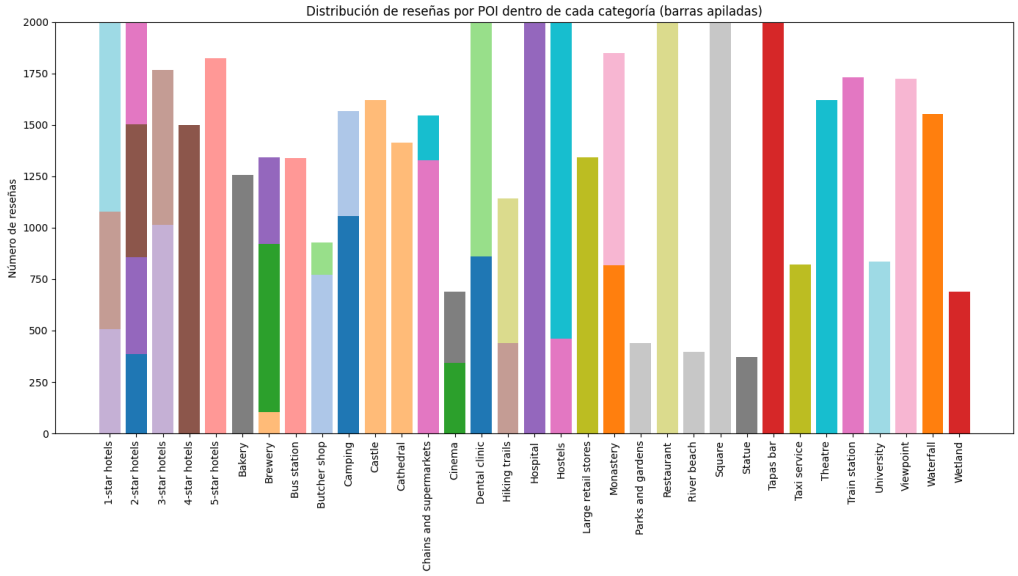

Figura C.2: Distribución de reseñas por POI y categoría

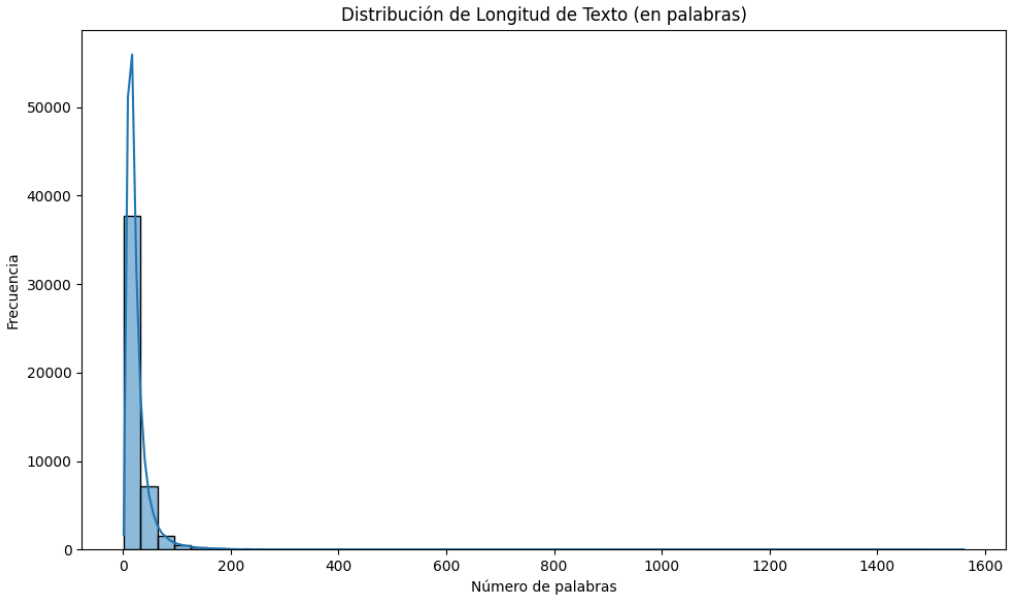

Figura C.3: Distribución de longitud de texto

Entrenamiento del modelo

En esta fase se crea la red neuronal y se entrena el modelo con los datos preprocesados. Se utiliza el modelo preentrenado de BERT en español y se entrena con los datos de las reseñas.

Generación de predicciones

Una vez entrenado el modelo, se utiliza para generar predicciones sobre las reseñas. Para ello, se realiza nuevamente una extracción y preprocesado de los recursos y reseñas que se quiera analizar y se pasan por el modelo obtenido para predecir el tipo de recurso. Finalmente, se obtiene un fichero .sql que se carga a la base de datos en la nube.

Sistema de recomendación

Tras la generación de predicciones, se utiliza nuevamente BERT para generar embeddings de los textos de las reseñas y junto a las coordenadas de los recursos se crean varios clusters que agrupan recursos similares. Nuevamente se obtiene un fichero .sql que se carga a la base de datos en la nube.

Análisis y visualización de resultados

Los resultados obtenidos tanto en la fase de predicción como de recomendación se muestran en el cuadro de mando de Power BI. Se lee la base de datos en la nube y se muestran lás métricas correspondientes.

C.4. Diseño procedimental

En esta sección se describen los procedimientos de cada script para realizar las diferentes tareas del proyecto. A continuación se muestran diferentes diagramas de secuencia que ilustran el flujo de trabajo de cada fase del proyecto.

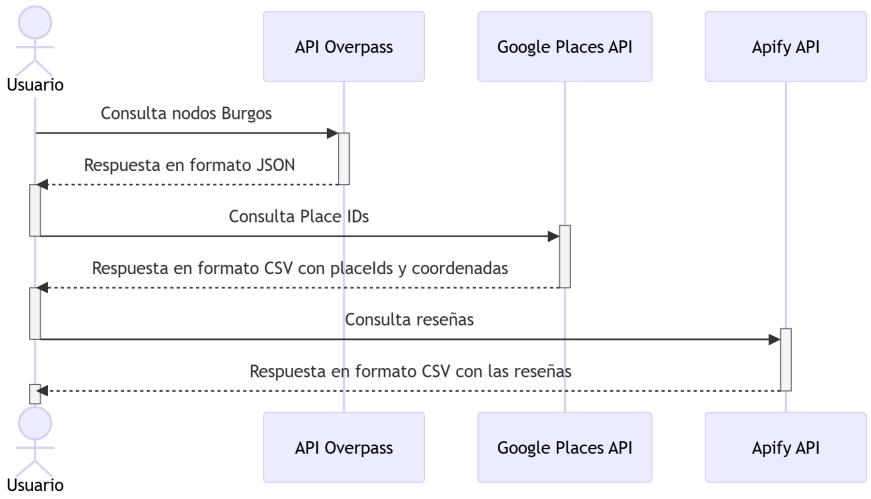

Figura C.4: Diagrama de secuencia de la extracción de datos

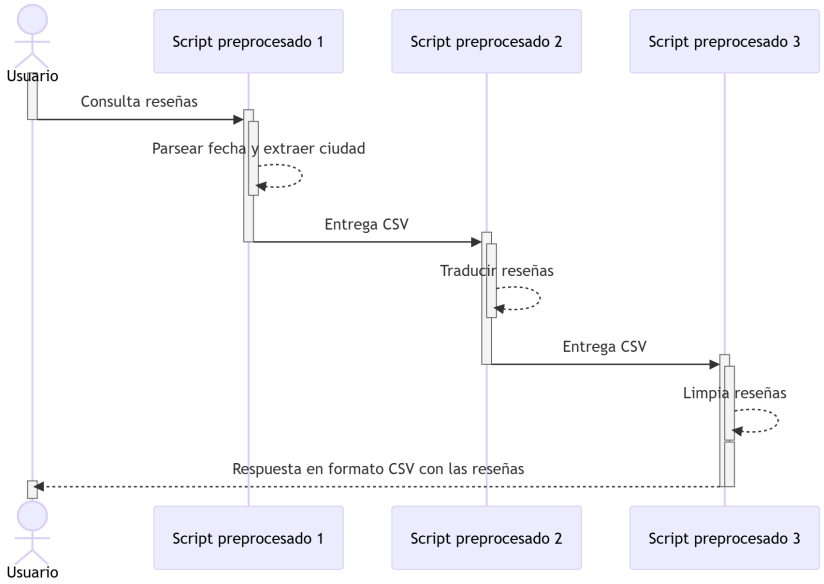

Figura C.5: Diagrama de secuencia del preprocesado de datos

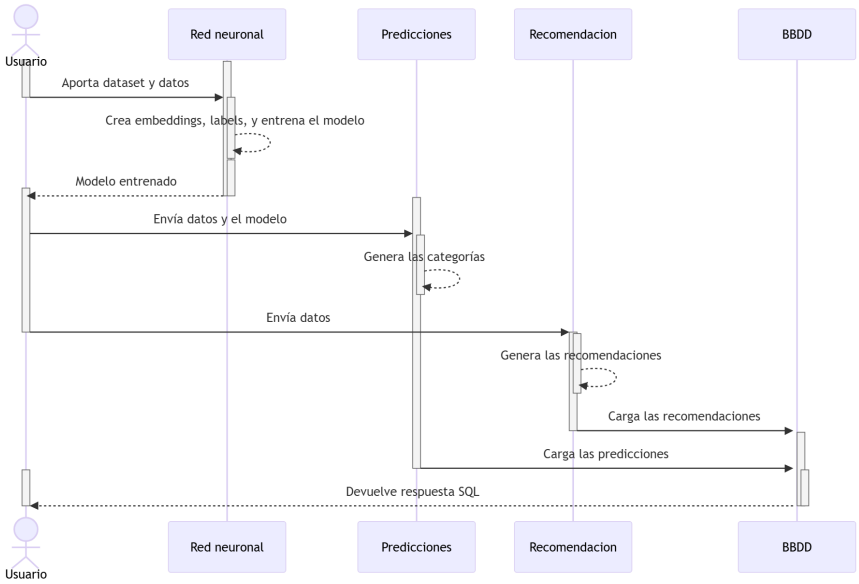

Figura C.6: Diagrama de secuencia de la red neuronal de datos

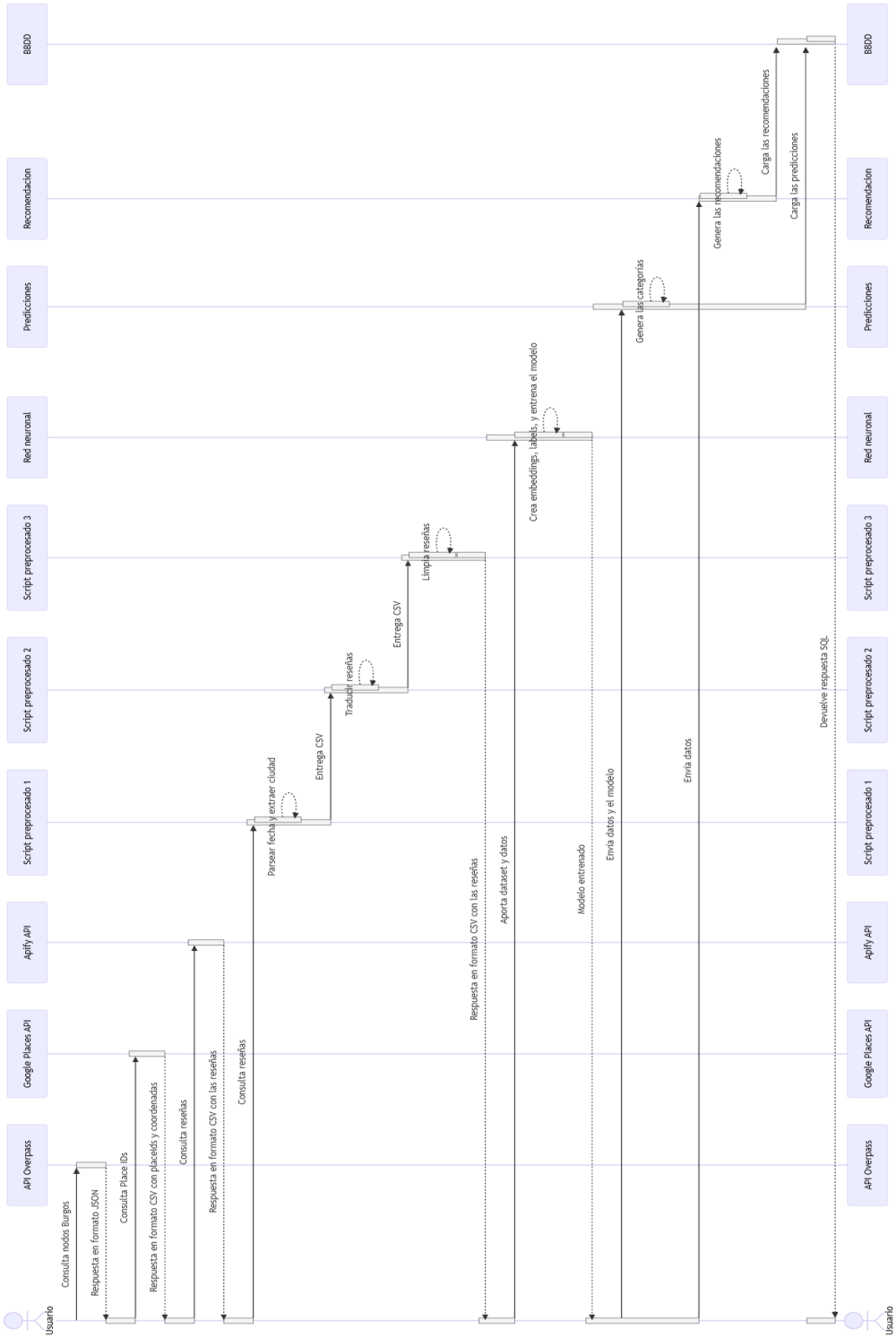

Figura C.7: Diagrama de secuencia completo

Apéndice D

Documentación técnica de programación

D.1. Introducción

En este anexo se presentan los aspectos relacionados con la estructura, instalación y ejecución del proyecto desde un punto de vista técnico. Se incluyen detalles relacionados a la estructura del repositorio que contiene el código fuente del proyecto o los pasos a seguir para ejecutar el mismo.

D.2. Estructura de directorios

Para alojar los ficheros fuente de este proyecto se ha creado un repositorio en GitHub[6]. Se ha tratado de seguir una estructura de directorios que permita una fácil comprensión de cada parte del proyecto. A continuación se explica la estructura de directorios del proyecto y su contenido:

- **.scannerwork/**: Contiene los ficheros de configuración del escáner de SonarQube utilizado para el análisis de código de forma local.
- **data/**: Contiene ficheros de datos con información sobre puntos de interés en Burgos.
  - **burgos_pois.json**: Fichero JSON con los nodos extraídos de la OpenStreetMaps a partir de Overpass.
  - **pois_details.csv**: Fichero CSV con las coordenadas y placeID de los puntos de interés obtenidos a partir del JSON anterior.

- **docs/**: Contiene todo lo utilizado para generar la memoria y los anexos.
  - **img/**: Contiene las imágenes utilizadas en la memoria y anexos.
  - **tex/**: Contiene los ficheros LaTeXque componen la memoria y los anexos.

  Además, incluye los ficheros de bibliografía en formato .bib y la memoria y anexos en formato PDF.
- **powerbi/**: Contiene los ficheros relacionados al cuadro de mando de PowerBI.
  - **dashboards/**: Contiene el fichero .pbix del cuadro de mando.
  - **icons/**: Contiene los iconos utilizados en el cuadro de mando, extraídos de icons8.[8]
  - **themes/**: Contiene el tema[7] utilizado en el cuadro de mando.
- **model/**: Contiene los ficheros relacionados con el modelo obtenido tras el entrenamiento de la red neuronal.
  - **bert_tokenizer.pkl**: Fichero con el tokenizer obtenido tras el entrenamiento.
  - **label_encoder.pkl**: Fichero con el encoder de las etiquetas de clasificación.
  - **tfidf_vectorizer.pkl**: Fichero con los vectorizers obtenidos tras el entrenamiento.
  - **review_model_combined_espanol.keras**: Fichero con el modelo entrenado.
- **sql/**: Contiene los ficheros SQL utilizados para inicializar la base de datos azure.
  - **poi_review_inserts.sql**: Fichero SQL con las tablas POI, reviews y algunos inserts.
  - **categories.sql**: Fichero SQL con los distintos niveles y sus categorías.
- **src/**: Contiene el código fuente del proyecto.
  - **dataExtraction/**: Contiene los scripts de Python utlilizados para la extracción de puntos de interés.

    - **OverpassExtraction.py**: Script de Python que utiliza la API de Overpass Turbo para extraer los nodos de OpenStreetMaps de puntos de interés de Burgos.
    - **PlaceIdExtraction.py**: Script de Python que utiliza la API de Google Places para obtener el placeID de los puntos de interés extraídos del script anterior.
    - **ApifyReviewExtraction.py**: Script de Python que utiliza la API de Apify para extraer las reseñas de los puntos de interés.
  - **preprocessing/**: Contiene los scripts de Python utilizados para el preprocesamiento de los datos.
    - **processReviews.py**: Script de Python que procesa las reseñas obtenidas de los ficheros CSV de Google Maps Review Scraper. No es necesario si se usa el script anterior de Apify.
    - **translateReviews.py**: Script de Python que traduce las reseñas que no estén en español y crea un campo 'text_original' con el texto original.
    - **cleanReviews.py**: Script de Python que limpia las reseñas eliminando saltos de línea y reseñas cortas.
    - **main.py**: Script de Python que ejecuta todo el pipeline de scripts anterior (solo los de preprocesamiento).
    - **mainApify.py**: Script de Python que ejecuta todo el pipeline de scripts anterior (solo los de preprocesamiento) sin el primer paso ya que no es necesario si se usa el script de Apify.
  - **finalInserts.ipynb**: Notebook de Jupyter para entrenar la red neuronal, realizar la clasificación de los puntos de interés y el sistema de recomendación.
  - **neuralNetwork.py**: Script de Python que contiene la red neuronal ya integrada en el notebook anterior.
  - **trainingDataset.csv**: Fichero CSV con el dataset creado y utilizado para el entrenamiento de la red neuronal.
- **README.md**: Fichero de texto markdown que contiene una descripción de como ejecutar el proyecto y sus requisitos.
- **LICENSE**: Fichero de texto que contiene la licencia del proyecto.
- **requirements.txt**: Fichero de texto que contiene las dependencias del proyecto.

- **docker-compose.yml**: Fichero de configuracion de Docker utilizado para levantar SonarQube.
- **sonar-project.properties**: Fichero de configuración de SonarQube para poder usar el escáner.

D.3. Manual del programador

En este apartado se explica todo lo necesario para poder ejecutar el proyecto y sus scripts. Es necesario seguir estas indicaciones antes de continuar con el proceso de instalación y ejecución del proyecto.

Entorno de desarrollo

Este proyecto ha sido desarrollado con diferentes programas y tecnologías detallados a continuación.

- **Python 3.10**: Lenguaje de programación utilizado para el desarrollo y ejecución de los scripts de extracción y preprocesamiento de datos.
- **Google Colab**: Entorno de desarrollo ofrecido por Google utilizado para ejecutar el fichero .ipynb sobre una GPU con el objetivo de acelerar su ejecución.
- **SQL Server Management Studio:** Herramienta ofrecida por Microsoft para la gestión de la base de datos SQL. Permite realizar consultas tras introducir las credenciales correspondientes.
- **Power BI Desktop**: Herramienta de Microsoft utilizada para crear el cuadro de mando. Permite integración con la base de datos.
- **Git**: Sistema de control de versiones utilizado para hacer cambios en el repositorio de GitHub.
- **Visual Studio Code**: Editor de código utilizado para el desarrollo de los ficheros fuente.
- **LaTeX y MikTeX**: Herramientas utilizadas para la generación de la memoria y anexos del proyecto.

Además, se han utlizado dos APIs para la extracción de datos a partir de una API Key que se puede obtener en sus páginas web:

- **Google Places API**: API utilizada para extraer los placeId de los puntos de interés de Burgos. No es necesario reutilizarla inmediatamente ya que ya han sido extraídos los puntos de interés.
- **Apify API**: API utilizada para extraer las reseñas de los puntos de interés. Es necesaria una Key para utilizar el fichero `ApifyReviewExtraction.py` y se puede obtener en su página web.

D.4. Compilación, instalación y ejecución del proyecto

En este apartado se describe el proceso completo de instalación y ejecución del proyecto. Cabe destacar que no es necesaria la completa ejecución de todos los scripts para poder ejecutar el proyecto.

Instalación

El primer requisito fundamental para la correcta ejecución del proyecto es tener instalado Python 3.10 o superior. Además, es necesario instalar las dependencias encontradas en el fichero `requirements.txt` que se encuentra en la raíz del proyecto. Estas dependencias se pueden instalar con el siguiente comando:

```
pip install -r requirements.txt
```

Ya vienen definidas las versiones de las dependencias necesarias. El fichero está dividido en dos partes, una para las dependencias de los scripts de Python y otra para las del notebook de Jupyter que he ejecutado en Google Colab. Estas últimas no son necesarias si se ejecuta el código allí. Por el contrario, si se ejecuta el notebook en local, es necesario instalarlas también. Para ello solo hay que eliminar el '#' del principio de las líneas del segundo bloque del fichero y ejecutar el comando anterior.

Ejecución

Todos los scripts explicados a continuación se pueden ejecutar desde la terminal siempre que nos encontremos en su directorio con el comando:

```
python3 <nombre_del_script.py>
```

Extracción de datos

Pipeline de extracción de datos

Vía Apify

Para la extracción de datos se han desarrollado varios scripts en Python que de ejecutarse ordenadamente permiten obtener las reseñas de los POIs de los municipios de la provincia de Burgos teniendo en cuenta la limitación de reseñas según el plan de pago de Apify.

El primer script a ejecutar es `OverpassExtraction.py`, que se encarga de extraer los nodos, vías y relaciones de los municipios de la provincia de Burgos. Este script utiliza la API de Overpass para obtener los límites de los municipios y guardarlos en un fichero JSON. Este script funciona a través de una consulta en Overpass QL. Ya que no todos los nodos de que ofrece OpenStreetMaps (OSM) son puntos de interés en Google Maps y por lo tanto no tienen reseñas, se eliminan en la consulta para reducir el tamaño del fichero JSON. Algunos de los nodos que se eliminan son papeleras, bancos, plazas de parking individuales, etc. El parámetro `admin_level` se utiliza para determinar el nivel de administración del área de la consulta. Se establece en 6 ya que es el nivel provincial lo que permite buscar dentro de los límites de la provincia de Burgos.

El segundo script a ejecutar es `PlaceIdExtraction.py` que se encarga de extraer los placeIds de los puntos de interés (POIs) obtenidos en el fichero JSON generado por el primer script. Este script utiliza la API de Google Places para obtener los placeIds de los POIs y guardarlos en un fichero CSV. Para ello se utiliza la función nearbySearch de la API de Google Places que permite buscar lugares cercanos a unas coordenadas dadas. Estas coordenadas se encuentran en el fichero JSON generado anteriormente por lo que basta con recorrer el fichero y realizar una búsqueda para cada uno de los POIs.

El tercer script a ejecutar es `ApifyReviewExtraction.py` que se encarga de extraer las reseñas de los POIs obtenidos en el fichero CSV generado por el segundo script. Este script utiliza la API de Apify para extraer las reseñas de los POIs y guardarlas en un fichero CSV. Para ello se utiliza el Google Maps Reviews Scraper de Apify que permite extraer las reseñas utilizando los placeIds obtenidos anteriormente. Este scraper devuelve una gran cantidad de campos de información, en mi caso he decidido filtrar solo los más relevantes para el proyecto como pueden ser la valoración o el nombre del punto de interés. También permite determinar el número de

reseñas a extraer por cada placeId. En caso de querer extraer todas hay que escribir 99999.

Las Keys necesarias para su ejecución se deben introducir en el fichero `.env` que se encuentra en la raíz del proyecto.

Vía Google Maps Review Scraper

Este scraper, disponible en GitHub, permite extraer reseñas a partir de una consulta mediante scraping. Devuelve varios ficheros CSV con multitud de campos. Su instalación viene detallada en el propio repositorio.[1]

Pipeline de preprocesamiento de datos

Este pipeline está formado por tres scripts de Python:

El primer script es `processReviews.py` que se encarga de procesar los ficheros CSV obtenidos del Google Maps Review Scraper para obtener un único fichero CSV con las reseñas y los POIs. Este script no es necesario si se ha utilizado el scraper de Apify ya que este último ya genera un fichero CSV con las reseñas y los POIs.

El segundo script es `translateReviews.py` que se encarga de detectar el idioma de las reseñas y traducirlas al español si es necesario. Para ello se utiliza la API de Google Translate que permite detectar el idioma de un texto y traducirlo a otro idioma. Este script también crea un campo `text_original` que contiene el texto original de la reseña sin traducir. Esto es útil para no perder la reseña original y poder compararla con la traducción.

El tercer script es `cleanReviews.py` que se encarga de limpiar las reseñas eliminando saltos de línea y reseñas con menos de 15 palabras. Esto es útil para garantizar que las reseñas contengan información suficiente para el análisis posterior.

Dependiendo de si se ha utilizado el scraper de Apify o el Google Maps Review Scraper, hay dos scripts principales que ejecutan el pipeline correspondiente:

- `main.py`: Este script ejecuta el pipeline de preprocesamiento de datos utilizando los ficheros CSV obtenidos del Google Maps Review Scraper.
- `mainApify.py`: Este script ejecuta el pipeline de preprocesamiento de datos utilizando los ficheros CSV obtenidos del scraper de Apify.

La única diferencia entre ambos es que en el script de Apify se omite el primer script de preprocesado.

El script `main.py` cuenta con varios parámetros a tener en cuenta que se definen en el propio código:

- `csv_reviews`: Fichero CSV de entrada con las reseñas obtenidas del Google Maps Review Scraper. Este fichero debe estar dentro de la carpeta `inputs/` que se encuentra al mismo nivel que el propio script.
- `csv_poi`: Fichero CSV de entrada con los POIs obtenidos del Google Maps Review Scraper. Este fichero debe estar dentro de la carpeta `inputs/` que se encuentra al mismo nivel que el propio script.
- `estado`: Provincia a escribir en el fichero de salida. Como todas las reseñas son de la provincia de Burgos se deja con ese valor.
- `max_reviews`: Número máximo de reseñas a extraer al finalizar el pipeline. Normalmente es irrelevante pero se ha utilizado para crear un dataset equilibrado.

Aprendizaje automático

Para ejecutar el fichero `finalInserts.ipynb` es necesario tener una cuenta de Google y acceder a Google Colab con ella. En este fichero se entrena la red neuronal y se obtienen las predicciones y las recomendaciones. Para entrenar la red neuronal, es necesario cargar el fichero `trainingDataset.csv` que se encuentra en la carpeta `src/` del proyecto. Además, hay que cargar el fichero `.env` que contiene las credenciales de la base de datos. Es necesario ejecutar las dos primeras celdas del notebook para descargar el connector de la base de datos y las dependencias necesarias. También es importante seleccionar el entorno de ejecución con GPU para acelerar el entrenamiento de la red neuronal o sus predicciones.

Si se desea saltar el entrenamiento de la red neuronal, se pueden cargar directamente los ficheros del modelo que se encuentran en la carpeta `model/` del proyecto. Se puede cargar un fichero .csv con las reseñas y los POIs para realizar las predicciones y recomendaciones. El formato recomendado es el obtenido tras ejecutar el pipeline de preprocesamiento de datos. Este pipeline devuelve como resultado un fichero `clean_reviews.csv` que es el que hay que cargar en Google Colab. Al ejecutar la celda de las predicciones, se generará un fichero `datos.sql` que se puede cargar en la base de datos para insertar las predicciones.

Respecto a las recomendaciones, no es necesario cargar ningún fichero ya que se generan en base al contenido de la base de datos. Al ejecutar su celda, se generará un fichero `recomendaciones.sql` que se puede cargar en la base de datos para insertar las recomendaciones.

En el notebook se incluye una función `cargar_sql_en_bd` y las llamadas correspondientes para cargar los ficheros de datos y recomendaciones. Esta función necesita tres parámetros:

- `ruta_sql`: Es la ruta del fichero SQL a cargar.
- `cursor`: Es el cursor de la conexión a la base de datos. Se crea durante la ejecución de las predicciones y/o recomendaciones.
- `conn`: Es la conexión a la base de datos. Se crea durante la ejecución de las predicciones y/o recomendaciones.

Cuadro de mando

Si se quiere ejecutar el cuadro de mando de Power BI, es necesario tener instalado Power BI Desktop. En caso de haberlo hecho, simplemente hay que abrir el fichero `dashboardFinal.pbix` que se encuentra en la carpeta `powerbi/dashboards/` del proyecto. Ahí se pueden ver los resultados finales. Para actualizar los datos tras cargarlos previamente en la base de datos desde Google Colab, solo hay que pulsar el botón de actualizar en la pestaña de Inicio en el menú superior. Dado que el cuadro de mando está conectado a la base de datos, se actualizarán los datos automáticamente y se podrán ver las predicciones y recomendaciones en el cuadro de mando solo en caso de que se hayan introducido las credenciales de Azure previamente.

Figura D.1: Cuadro de mando final

El cuadro de mando se encuentra publicado actualmente en PowerPages de forma privada debido a restricciones de permisos relacionados con las cuentas de la Universidad. Para publicarlo, hay que ir a archivo y seleccionar Publicar en la web. A continuación, se debe ir al portal de Power BI en el navegador y seleccionar el cuadro de mando publicado y en archivo, insertar informe, sitio web o portal. Esto generará una URL y un código HTML que se puede incrustar en una página web ya sea de PowerPages o de cualquier otro sitio web.

D.5. Pruebas del sistema

En relación a las pruebas del sistema, se han realizado varias pruebas manuales con el fin de verificar el correcto funcionamiento de los scripts y del cuadro de mando.

La prueba más importante ha sido la relacionada con la red neuronal. Para seleccionar los hiperparámetros, se ha realizado un random search con el fin de obtener los mejores hiperparámetros posibles.

Además, se ha probado con varios conjuntos de datos para comprobar como evolucionaban las métricas de la red neuronal. Esto ha llevado a la conclusión de intentar construir un dataset multiclase equilibrado con el fin

de que la red neuronal pueda aprender de todas las clases por igual. Para ello, se ha utilizado el parámetro `max_reviews` del script `main.py`.

Apéndice E

Documentación de usuario

E.1. Introducción

En este apéndice se presenta la documentación de usuario necesaria para que el usuario final pueda utilizar el sistema desarrollado en este proyecto. Este apéndice está enfocado al usuario final, que en principio no tiene conocimientos técnicos sobre el sistema y cuyo objetivo es usar el cuadro de mando final.

E.2. Requisitos de usuarios

El usuario final debe contar con varios requisitos para poder utilizar el cuadro de mando:

- Tener acceso a un navegador web moderno con conexión a Internet.
- Tener acceso a una cuenta de Power BI y su aplicación descargada para poder acceder al cuadro de mando.

E.3. Instalación

La 'instalación' del cuadro de mando es un proceso muy simple. Simplemente hay que abrir el fichero `dashboardFinal.pbix` con la aplicación de Power BI Desktop. Una vez abierto, el usuario podrá ver el cuadro de mando y navegar por él.

E.4. Manual del usuario

El cuadro de mando está dividido en varias páginas. El usuario puede navegar de dos formas diferentes a través de dos componentes diferentes.

Navegación por pestañas

PowerBI cuenta con un menú inferior con las pestañas del cuadro de mando. Simplemente haciendo clic sobre ellas se puede navegar entre ellas.

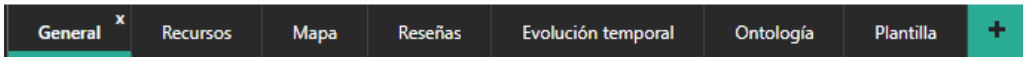

Figura E.1: Pestañas de navegación

Navegación por botones (sidebar)

También se ha implementado de forma manual una barra lateral con iconos para navegar entre las difernetes pestañas. Para usarlos, el usuario debe pulsar el botón CONTROL a la vez que hace clic en el icono deseado. Se muestra a continuación la barra lateral de forma apaisada. (En el cuadro de mando se encuentra a la izquierda en vertical).

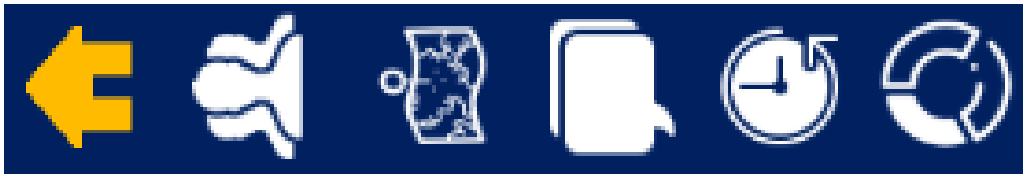

Figura E.2: Barra lateral

Filtros

El cuadro de mando cuenta con dos filtros principales basados en los tipos de recursos y municipios. Estos filtros están disponibles en todas las pestañas.

Figura E.3: Filtros de recursos y municipios

Métricas

Todas las gráficas del cuadro de mando son interactivas e intuitivas. El usuario puede clicar en ellas para filtrar de la misma forma que con los filtros principales. Las métricas principales del cuadro de mando son el TORI, el número de reseñas y la polaridad.

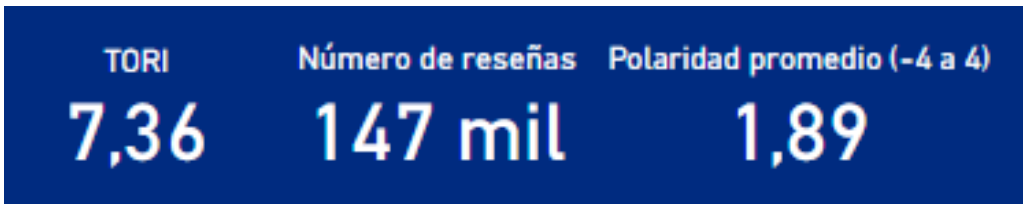

Figura E.4: Métricas principales

Borrado de filtros

Se cuenta con un botón para borrar los filtros aplicados en el cuadro de mando. Este botón se encuentra en la parte superior derecha y se acciona pulsando CONTROL a la vez que se hace clic sobre él.

Figura E.5: Botón de borrado de filtros

Ayuda

Finalmente, cada pestaña cuenta con una sección de ayuda a la derecha que explica las funcionalidades desarrolladas en cada página.

Páginas del cuadro de mando

A continuación se muestran dos ejemplos de páginas del cuadro de mando en el que se ven los aspectos anteriores.

Figura E.6: Página de recursos

Figura E.7: Página de evolución temporal

Apéndice F

Anexo de sostenibilización curricular

F.1. Introducción

En este apéndice se explica la relación que existe entre el proyecto y los Objetivos de Desarrollo Sostenible (ODS) y las competencias de sostenibilidad adquiridas durante su desarrollo. Para ello, se da una breve explicación de los ODS correspondientes y se relacionan con las tareas o funcionalidades realizadas en el proyecto.

Estos ODS fueron aprobados en 2015 por todos los miembros de la Organización de las Naciones Unidas (ONU) y tienen como objetivo acabar con la pobreza, mejorar la salud y la educación, reducir las desigualdades, garantizar la paz y prosperidad de la gente y del planeta e impulsar el crecimiento económico.[4]

F.2. Objetivos de Desarrollo Sostenible

El proyecto se centra en la sostenibilidad de los recursos turísticos de la provincia de Burgos, lo que implica una gestión responsable y sostenible de estos recursos.

ODS 5: Igualdad de género

Se proporcionan gráficas que permiten diferenciar las valoraciones y número de reseñas de los diferentes recursos dependiendo del género de los

usuarios. Esto permite identificar si hay alguna diferencia significativa en la valoración de los recursos turísticos entre hombres y mujeres, lo que puede ayudar a mejorar la oferta turística y adaptarla a las necesidades de ambos géneros. También puede ayudar a identificar los recursos más valorados por un género u otro y que puede llevar a fomentar la visibilidad de los colectivos poco representados.

ODS 9: Industria, innovación e infraestructura

El desarrollo de este proyecto ha implicado el uso de tecnologías innovadoras como una red neuronal en relación al aprendizaje automático. Además, la explotación de los datos obtenidos de forma pública permite mejorar la infraestructura turística de la provincia de Burgos.

ODS 11: Ciudades y comunidades sostenibles

Este proyecto contribuye a la sostenibilidad de las ciudades y comunidades al proporcionar información sobre los recursos de la provincia de Burgos. A través del cuadro de mando, se puede identificar y analizar las zonas con un determinado tipo de recurso lo que permite detectar zonas mal valoradas o con pocos recursos que podrían ser mejoradas.

F.3. Competencias de sostenibilidad adquiridas

A través del desarrollo de este proyecto, he adquirido competencias de sostenibilidad definidas por la Conferencia de Rectores y Rectoras de las Universidades Españolas (CRUE)[2].

SOS1 - Competencia en la contextualización crítica del conocimiento estableciendo interrelaciones con la problemática social, económica y ambiental, local y/o global

Durante el desarrollo del proyecto se ha prestado atención al contexto social y territorial de la provincia de Burgos, entendiendo cómo la distribución de los puntos de interés (POIs) y la interacción de los ciudadanos con estos espacios a través de reseñas puede reflejar desigualdades económicas, concentración de servicios y posibles carencias estructurales.

SOS2 - Competencia en la utilización sostenible de recursos y en la prevención de impactos negativos sobre el medio natural y social

El proyecto ha utilizado datos públicos de Google Maps, lo que implica un uso sostenible de los recursos digitales disponibles. Además, en todo momento se ha tratado de optimizar y minimizar el impacto ambiental del proyecto, evitando el uso de recursos innecesarios y fomentando la reutilización de datos existentes.

SOS4 - Competencia en la aplicación de principios éticos relacionados con los valores de la sostenibilidad en los comportamientos personales y profesionales

En relación a la ética, se ha procurado garantizar la transparencia en el uso de los datos y en la presentación de los resultados, evitando cualquier tipo de manipulación o tergiversación de la información.

F.4. Conclusiones

En conclusión, he aprendido a relacionar el proyecto con los ODS y a reflexionar sobre la sostenibilidad de los recursos turísticos de la provincia de Burgos. He adquirido competencias de sostenibilidad que me permiten entender la importancia de la sostenibilidad en el desarrollo de proyectos y en la gestión de recursos turísticos.

Bibliografía

[1] Cheetan11dev. omkarcloud/google-maps-scraper. https://github.com/omkarcloud/google-maps-scraper, 2025. [Internet; último acceso 2-Julio-2025].

[2] CRUE. Directrices para la introducción de la sostenibilidad en el curriculum. https://www.crue.org/wp-content/uploads/2020/02/Directrices_Sosteniblidad_Crue2012.pdf, 2014. [Internet; último acceso 3-Julio-2025].

[3] Datstrats. ¿es legal el scraping en españa? https://datstrats.com/blog/scraping-es-legal-espana/, (s.f). [Internet; último acceso 2-Julio-2025].

[4] Organización de las Naciones Unidas. The 17 goals | sustainable development. https://sdgs.un.org/es/goals, (s.f). [Internet; último acceso 3-Julio-2025].

[5] Apify Developers. Apify pricing. https://apify.com/pricing, 2025. [Internet; último acceso 29-Abril-2025].

[6] C. A. Núñez Duque. cnunez1/tfg-digitalizacionboletinturismo. https://github.com/cnunez1/TFG-DigitalizacionBoletinTurismo, 2025. [Internet; último acceso 7-Julio-2025].

[7] FerArg. Re: Simplifica - otoño - autumn - microsoft fabric community. https://community.fabric.microsoft.com/t5/Themes-Gallery/Simplifica-OtoÃ±o-autumn/m-p/4700294, 2025. [Internet; último acceso 7-Julio-2025].

[8] Icons8. Iconos gratis, ilustraciones clipart, fotos y música. https://iconos8.es, 2025. [Internet; último acceso 7-Julio-2025].