UNIVERSIDAD EUROPEA DE MADRID

ESCUELA DE ARQUITECTURA, INGENIERÍA Y DISEÑO

MÁSTER UNIVERSITARIO EN

ANÁLISIS DE DATOS MASIVOS (BIG DATA)

TRABAJO FIN DE MÁSTER

Análisis y Comparativa de las Listas de Sancionados de la Unión Europea y Estados Unidos

Alberto Alonso Chana

CURSO 2021-2022

TÍTULO: Análisis y Comparativa de las Listas de Sancionados de la Unión Europea y Estados Unidos

AUTOR: Alberto Alonso Chana

TITULACIÓN: Máster Universitario en Análisis de Datos Masivos (Big Data)

DIRECTOR DEL PROYECTO: Jorge Luis Hita y José Javier Ruiz Cobo

FECHA: septiembre de 2022

RESUMEN

Este trabajo pretende mostrar un caso de uso real de los métodos y técnicas aplicadas en ingeniería de datos e inteligencia de negocio.

A través del estudio de las listas de sancionados financieros por parte de los servicios de seguridad de Estados Unidos y la Unión Europea, se construye todo el proceso de extracción, limpieza, transformación, integración, carga y representación de los datos.

Se eligieron este tipo de datos porque las sanciones internacionales están a la orden del día y por el reto que a priori entrañaba el procesar estas listas, las cuales presentaban una calidad de datos en crudo bastante baja.

Palabras clave:

Python: Lenguaje de programación interpretado de código abierto. Algunas de sus aplicaciones principales son la manipulación de datos estructurados y la implementación de algoritmos de Inteligencia Artificial.

DataFrame: Un DataFrame es una estructura de datos con dos dimensiones en la cual se puede guardar datos de distintos tipos (como caracteres, enteros, valores de punto flotante, factores y más) en columnas. Un DataFrame siempre tiene un índice (con inicio en 0) [18]. Esta estructura de datos es utilizada en Python importándola desde la librería pandas.

Jupyter Notebook: Los Jupyter Notebooks son una aplicación web de código abierto que permite crear y compartir documentos con código en vivo, ecuaciones, visualizaciones y texto explicativo [19]. Python es uno de los lenguajes que más a menudo se utiliza dentro de un notebook de Jupyter. La extensión del archivo es .ipynb.

Herramientas de visualización: Se tratan de aplicaciones que utilizan representaciones visuales y cálculos estadísticos para interpretar grandes volúmenes de datos. Son útiles en la toma de decisiones y para el desarrollo de la inteligencia de negocio. Ejemplos de estas herramientas son Power BI, Google Charts y Tableau.

ABSTRACT

This work aims to show a real use case of the methods and techniques applied in data engineering and business intelligence.

Through the study of the lists of financial sanctioned by the security services of the United States and the European Union, the entire process of extraction, cleaning, transformation, integration, loading and representation of data is built.

This type of data was chosen because international sanctions are the order of the day and because of the challenge that a priori involved in processing these lists, which presented a fairly low raw data quality.

Key words:

Python: Open source interpreted programming language. Some of its main applications are the manipulation of structured data and the implementation of Artificial Intelligence algorithms.

DataFrame: A DataFrame is a two-dimensional data structure in which data of different types (such as characters, integers, floating point values, factors, and more) can be stored in columns. A DataFrame always has an index (starting at 0) [18]. This data structure is used in Python by importing it from the pandas library.

Jupyter Notebook: Jupyter Notebooks are open source web applications that allows you to create and share documents with live code, equations, visualizations, and explanatory text [19]. Python is one of the languages most often used within a Jupyter notebook. The file extension is .ipynb.

Visualization Tools: These are applications that use visual representations and statistical calculations to interpret large volumes of data. They are useful in decision making and for the development of business intelligence. Examples of these tools are Power BI, Google Charts and Tableau.

Índice

Capítulo 1. INTRODUCCIÓN

1.1 Descripción del Problema

Las sanciones internacionales son medidas coercitivas que los Gobiernos y las entidades supranacionales aplican contra Estados, empresas o individuos que suponen una amenaza para la seguridad global. Abarcan restricciones económicas, diplomáticas, comerciales, militares e incluso deportivas, y sus objetivos son debilitar y reducir la capacidad de maniobra de estos actores sin hacer uso de la fuerza [1].

1.1.1 Contexto Histórico

Las sanciones se han utilizado como herramienta diplomática desde la Antigüedad. La primera conocida fue el embargo comercial de Atenas a Mégara en el 432 a. C. Posteriormente, durante el siglo XIX, el uso de sanciones se intensificó a partir del bloqueo continental que Napoleón decretó en 1806 con el fin de excluir cualquier intercambio con el Reino Unido, su rival por el control de Europa. No obstante, el auge definitivo de las sanciones internacionales llegó con el final de la Primera Guerra Mundial. Durante este período, la Sociedad de Naciones (SDN), predecesora de la ONU, fue fundamental en el intento de imponer medidas para boicotear a los países agresores. En los años noventa, la invasión de Kuwait por el Irak de Sadam Huseín y las guerras yugoslavas llevaron a un nuevo repunte de las sanciones internacionales [1].

El uso de las sanciones se ha consolidado en la comunidad internacional en las últimas décadas. Su adopción ha trascendido de la ONU a la **Unión Europea**, en su Política Exterior y de Seguridad Común, o a la Organización para la Seguridad y Cooperación Europea (OSCE), entre otras. A nivel estatal, **Estados Unidos** es el país que más ha utilizado este mecanismo, a través de leyes y órdenes ejecutivas. La Ley para la Democracia Cubana de 1992, por ejemplo, reguló el embargo a la isla [1].

1.1.2 Actualidad

Ya en 2022, Tanto Washington como Bruselas decidieron apostar por estas medidas para responder a la invasión rusa de Ucrania. Las sanciones, que incluyen el cierre del espacio aéreo europeo a Rusia o su desconexión parcial del sistema Swift de comunicaciones financieras, pretenden aislar la economía rusa. Las restricciones también alcanzan al ámbito deportivo, como el veto a Rusia en el Mundial de fútbol de Catar [1].

1.2 Motivación

Para un mejor control y seguimiento de las entidades sancionadas, tanto la Unión Europea como Estados Unidos elaboran unas listas que ponen a disposición de la ciudadanía a través de internet. Las personas, grupos y compañías que aparecen en estas listas tienen sus activos bloqueados y generalmente se prohíbe a las empresas europeas y estadounidenses tratar con ellos.

Los avances en análisis y procesamiento de datos pueden aportar nuevos enfoques a la forma en que estas sanciones son interpretadas y aplicadas:

- Mediante algoritmos de Machine Learning de aprendizaje no supervisado, se pueden conocer patrones ocultos de los datos que de otra manera pasarían desapercibidos. El aprendizaje supervisado permite conocer la información faltante de algunos registros a partir de los registros en los que sí existe esa información.

- Las herramientas enfocadas a extraer información a partir de los datos mediante la generación de dashboards son cada vez más avanzadas. Estas son capaces de ingerir grandes cantidades de datos y mantenerlos siempre actualizados, además de presentarlos de forma dinámica e interconectada.

Estos avances benefician a las empresas susceptibles de tener entre sus clientes a entidades sancionadas. Existen diversos motivos por los que una compañía querría conocer cuáles son sus clientes sancionados: mantener la reputación, evitar el pago de multas, cumplir con la normativa para la no financiación del terrorismo, etc. [2]

Este trabajo tiene como objetivo tanto realizar un proceso de investigación de los datos en busca de relaciones e informaciones desconocidas, como de generar valor a las empresas que pudieran beneficiarse del seguimiento de los sancionados mediante dashboards generados con herramientas de visualización accesibles y fáciles de usar. Esto con el objetivo de que cualquier miembro de la compañía en cualquier departamento, incluido el equipo directivo, pueda disponer fácilmente de la información e interpretarla sin necesidad de ser experto en el campo.

1.3 Conceptos Clave

En esta sección se van a describir de forma teórica los métodos y técnicas necesarios en el desarrollo del trabajo (proceso ETL, limpieza de datos, integración). También se definirán aquellos conceptos de programación que más tarde serán aplicados de forma práctica.

1.3.1 Técnicas de Procesamiento de Datos

a) **Proceso ETL:** Este proceso garantiza que los datos tengan la cohesión necesaria para su posterior implementación en las aplicaciones y sistemas. La palabra ETL corresponde a las siglas en inglés de Extraer, Transformar y Cargar.

El proceso ETL garantiza que los datos se puedan mover entre múltiples fuentes, los datos sean reformateados y limpiados cuando sea necesario, sean cargados en otras plataformas (base de datos, data mart, data ware-house, etc.) o puedan ser empleados en otro sistema operacional [7].

A continuación, se describen las claves de cada fase del proceso ETL [7]:

Proceso de extracción: Consiste en tomar los datos desde los sistemas de origen.

- Analizar los datos extraídos obteniendo un chequeo.
- Interpretar este chequeo para verificar que los datos extraídos cumplen la pauta o estructura que se esperaba. Sino fuese así, los datos deberían ser rechazados.
- Convertir los datos a un formato preparado para iniciar el proceso de transformación. Para evitar este impacto y sus consecuencias, en sistemas grandes, las operaciones de extracción suelen programarse en horarios o días donde la interferencia con el sistema y su uso sea nula o mínima.

Proceso de transformación: Los datos extraídos son transformados para luego poder ser cargados. Para asegurar la eficacia de este proceso, hay que asegurarse de que las alertas y cambios producidos durante esta fase sean:

- Declarativas.
- Independientes.
- Claras.
- Inteligibles.
- Con una finalidad útil.

Proceso de carga: Los datos transformados son cargados en el programa, herramienta o aplicación de destino. Las acciones tomadas en este proceso dependerán del tipo de plataforma de destino.

b) **Limpieza de datos:** Se trata del proceso de corregir o eliminar datos en formato incorrecto, duplicados o perdidos dentro de un gran conjunto de datos [8].

Sirve para analizar, identificar y corregir datos en bruto que están desordenados, equivocados y mal procesados. El proceso de limpieza de datos trata de completar los valores faltantes, corregir errores y determinar si toda la información está en las filas y columnas correctas [8].

Según Tableau, se pueden implementar los siguientes pasos básicos para empezar a limpiar los datos [8]:

- **Eliminar los datos duplicados o irrelevantes**: Cuando se realiza la combinación de datos de distintos sitios o se reciben datos de clientes de diferentes páginas de registro, existe una gran posibilidad de crear datos duplicados.

- **Corregir los errores estructurales**: Los errores estructurales ocurren cuando se mide o se transfiere datos (errores tipográficos o nomenclaturas extrañas). Estas inconsistencias pueden causar categorías mal etiquetadas.

- **Filtrar valores atípicos no deseados:** A menudos aparecen observaciones únicas que no parecen encajar dentro de los datos a analizar. Puede tratarse de una entrada de datos incorrecta. Sin embargo, el hecho de que exista algún valor atípico no siempre significa que sea incorrecto.

- **Manejar los valores faltantes:** Al realizar limpieza de datos no se puede ignorar los valores faltantes, ya que muchos algoritmos no aceptan valores vacíos. Según la situación, estos valores pueden ser eliminados, sustituidos por otros valores en base a otras observaciones o modificarse la forma en que se utilizan para navegar de forma efectiva. Al proceso de sustitución de valores faltantes se le conoce con el nombre de imputación.

- **Validar y controlar la calidad**: ¿Tienen sentido los datos? ¿Los datos siguen las reglas apropiadas de cada campo? ¿Los datos obtenidos prueban o refutan una teoría importante? ¿Puedes encontrar tendencias en los datos que ayuden a tu organización? ¿Conseguiste datos de calidad?

c) **Integración de datos:** Este proceso permite combinar datos de diferentes orígenes y formato en un único lugar.

   La integración de datos combina y alinea cada conjunto de datos procedentes de distintas fuentes para alcanzar la integración necesaria en las tareas y procesos que se vayan a realizar [9]

1.3.2 Conceptos Informáticos

a) **Expresiones regulares o regular expressions (regex):** Las regex son cadenas de caracteres basadas en reglas sintácticas que permiten describir secuencias de caracteres [10].

   Un ejemplo del uso de expresiones regulares es la función de buscar y reemplazar en editores de texto, la cual permite buscar secuencias de caracteres en los textos y poderlas reemplazar por otra secuencia de nuestra elección [10].

En este trabajo, las expresiones regulares son utilizadas para identificar patrones en todos los valores de una misma columna de datos y poder aplicar cambios sobre ellos que de otra forma sería imposible hacer.

b) Machine Learning & Deep Learning: El Machine Learning (ML) es la ciencia de programar ordenadores para que aprendan a partir de datos. Es el campo de estudio que da a los ordenadores la capacidad de aprender sin ser programados de manera explícita [11].

El Deep Learning forma parte del Aprendizaje Automático o Machine Learning. Los algoritmos de Deep Learning se diferencian del resto de algoritmos de Inteligencia Artificial en que buscan imitar el cerebro humano y la conexión entre neuronas [12].

Existen lenguajes de programación con acceso a librerías especializadas en Machine Learning y Deep Learning. Es el caso de las librerías scikit-learn y keras de Python.

c) Aprendizaje no supervisado: Se trata un tipo de aplicación específica de Machine Learning. Se diferencia del aprendizaje supervisado en que los datos no están etiquetados. Esto es, no existe un set de entrenamiento con datos etiquetados a partir de los cuales se pueda conocer el valor de la etiqueta del resto de datos.

Existen diversas aplicaciones de los algoritmos de aprendizaje no supervisado [11]:

- **Agrupamiento:** Se utiliza para detectar grupos de datos similares. Algunos ejemplos son: K-Medias, DBSCAN y análisis de agrupamiento jerárquico.

- **Visualización y reducción de la dimensionalidad:** Los algoritmos de visualización también son buenos ejemplos de algoritmos de aprendizaje no supervisado: se les introduce gran cantidad de datos complejos y sin etiquetar y generan representaciones 2D o 3D de los datos que pueden trazarse con facilidad.

  Una tarea relacionada es la reducción de la dimensionalidad. El objetivo es simplificar los datos sin perder demasiada información. Una manera de hacerlo es fusionar varias características correlacionadas en una.

  Algunos de estos algoritmos son: Análisis de componentes principales con o sin *kernel* (k-PCA/ PCA), LLE (*Locally Linear Embedding*) y técnica t-SNE (*t-Distributed Stochastic Neighbor Embedding*).

- **Detección de anomalías:** Se muestran al sistema instancias normales en su mayoría durante el entrenamiento, así que aprende a reconocerlas; después, cuando ve una nueva instancia, puede decir si parece una normal o si es probable que sea una anomalía: SVM de una clase, *isolation forest.*

- **Reglas de asociación:** Tienen como objetivo explorar cantidades enormes de datos y descubrir relaciones interesantes entre los atributos: Apriori, Eclat.

d) Programación orientada a objetos (POO): En un esquema de programación orientada a objetos la **clase** contiene la definición de las características de un modelo. Las características definidas en la clase las llamamos **propiedades** y las funcionalidades asociadas a la clase con los **métodos**. De una clase se pueden crear cualquier número de **objetos** de esa clase, que son los elementos concretos creados a partir de una clase [16].

En resumen, al definir una clase con ciertas propiedades se crea un objeto específico de esa clase sobre el que se pueden aplicar métodos que nos lleven a resultados.

En este trabajo, la POO se utilizará en Python principalmente cuando utilicemos la librería scikit-learn. Esta librería tiene clases predefinidas centradas principalmente en el preprocesamiento de datos y aplicación de modelos de Machine Learning. La descripción de las clases de scikit-learn y ejemplos de código se pueden encontrar en su web oficial:

https://scikit-learn.org/stable/modules/classes.html

La diferencia con la definición de POO dada es que en la web a las propiedades se las llama **parámetros** y aparece un nuevo elemento que devuelve información del objeto, los **atributos**, similares a los métodos.

1.4 Estructura del proyecto

Capítulo 1: Se comienza haciendo una descripción general del problema que se va a abordar y de sus antecedentes. Seguidamente, se explica la motivación del trabajo y los objetivos generales que se pretenden alcanzar. Por último, se introducen los conceptos fundamentales en el desarrollo del trabajo.

Capítulo 2: Se hace una memoria del trabajo donde se da una descripción detallada del procedimiento seguido para cumplir con los objetivos. En este capítulo se documentan las técnicas, tareas realizadas, imprevistos y como se solucionaron, resultados y valoraciones del trabajo. Las ideas aquí expresadas se apoyan en figuras que muestran tablas y diagramas.

Capítulo 3: Se exponen las conclusiones a las que se ha llegado y se proponen futuras líneas de trabajo que pudieran ampliar y mejorar el presente proyecto.

Capítulo 2. Memoria Técnica

Los ficheros, scripts, tablas, dashboards y demás documentos empleados en este trabajo, incluida esta memoria, están publicados en el siguiente repositorio de Github:

https://github.com/Chanalber/TFM.git

A continuación, se exponen las tareas realizadas durante el desarrollo del trabajo y la forma en que fueron abordadas. En este capítulo se tratarán los problemas, cambios e imprevistos en la consecución de estos objetivos para finalmente presentar los resultados a los que se ha llegado.

2.1 Procesado de Datos

En este apartado se explicará cómo se llevó a cabo la depuración de los datos de Estados Unidos y la Unión Europea y su posterior integración. Una vez finalizado el proceso, se tiene una tabla única con todos los sancionados y en un formato adecuado para ser utilizadas por los modelos ML y de análisis visual de los datos.

El proceso ETL (Extract, Transform, Load) engloba el conjunto de técnicas aplicadas sobre los datos en crudo con objeto de obtener un dataset depurado y listo para aportar valor mediante diversos métodos: entrenamiento de modelos de Machine Learning, generación de visualizaciones de Inteligencia Empresarial, etc.

La integración de datos permite seleccionar datos de distintas fuentes y combinarlos en una misma estructura.

Durante el procesado, se ha combinado el proceso ETL con técnicas de limpieza de datos y otros métodos computacionales para depurar los datos. Una vez hecho esto, se unieron los datos de la Unión Europea y de Estados Unidos en una misma tabla.

Aunque este apartado se ha intentado dividir en varios subapartados en función de las técnicas de procesamiento aplicadas, no siempre se puede separar cada técnica como una entidad independiente. Pues puede ocurrir que una técnica dé forma a otra y funcionen conjuntamente.

2.1.1 Extracción y Selección de Datos

El notebook de Jupyter para código Python ha sido la aplicación utilizada en la carga de los datos de las listas de sancionados. Se generó un script de nombre *01_extraer_datos.ipynb* en la ubicación *TFM/Código y datos/* con los siguientes propósitos generales:

- Acceder a las listas de sancionados de EEUU y la UE vía URL y cargar los datos en Python transformándolos a formato de DataFrame.

- Almacenar los datos en crudo en formato .csv en la ubicación *TFM/Código y datos/UE_EEUU_data_crudo* para tenerlos localizados en caso de que se quieran revisar. Cada vez que este proceso es ejecutado, se borran los archivos antiguos del directorio local y se vuelven a cargar desde la web.

- Selección de los datos de los DataFrame que sean de interés. Habrá dos tablas para la lista de EEUU relacionadas por clave y una para la UE. Los datos seleccionados se centran en tres dimensiones de la información de sancionados: **Nombre, ubicación y tipo de sancionado.**

- Pasar los DataFrame con los datos cargados y seleccionados a formato .csv y almacenarlos en la ubicación *TFM/Código y datos/datos_entrada_script_02.* Esto se hace para que los datos que vayan a transformarse con el script 02 (del que se hablará más adelante) puedan ser tomados directamente desde esta ubicación siempre que se quiera, sin ser necesario ejecutar antes el script 01. De este modo, solo será necesario ejecutar el script *01_extraer_datos.ipynb* antes del script 02 en caso de que se quiera tener la última versión de los datos de entrada al script 02.

Lista de Estados Unidos

Para cargar en el script de Python las url's con las listas de sancionados de EEUU se utiliza wget.download() dentro de un bucle for que se ejecuta una vez por cada url. Cada vez que este proceso es ejecutado, se borran los archivos antiguos en el directorio local *TFM/Código y datos/UE_EEUU_data_crudo* y se vuelven a cargar desde la web [3].

Al principio se intentó cargar la lista de EEUU desde un único .xml con toda la información de los sancionados. Pero esto no fue posible porque, al pasarlo a DataFrame, los elementos anidados del .xml no permitían su correcta representación. Por ello, se optó por acceder a la misma información, pero dividida en varios .csv, que se encontraban en la misma página web que el .xml. De entre estos archivos .csv, los que llevan información relevante para su estudio son: *sdn.csv* y *add.csv.* Estos archivos se relacionan por una columna llamada 'uid' por lo que, tras ponerlos en formato DataFrame y seleccionar las columnas relevantes, se hace un inner join a las tablas.

Las columnas de los .csv no vienen etiquetadas con un nombre. Es necesario buscar el nombre correspondiente a cada columna en el archivo .xml con la información general y escribirlo.

```
df_eeuu.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 15287 entries, 0 to 15286
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   LastName  15287 non-null  object
 1   sdnType   8103 non-null   object
 2   Address   8250 non-null   object
 3   City      10059 non-null  object
 4   Country   12864 non-null  object
dtypes: object(5)
memory usage: 716.6+ KB
```

Figura 1: Información de metadatos de la tabla df_eeuu.

Una vez se tiene un único DataFrame para la lista de EEUU, al que se llama *df_eeuu*, se estudian los valores de cada variable en busca de posibles incidencias (Fig. 1). Se encuentra qué para la columna con la información del tipo de sancionado, llamada 'sdnType', hay demasiados valores nulos. El tipo de sancionado puede deducirse a partir del nombre del sancionado (columna de nombre 'LastName'). Para ello, se crea un bucle que devuelve el nombre de todos los sancionados cuya columna 'sdnType' es nula. Una vez hecho, se observa en la Figura 2 que la mayoría de sancionados se corresponden a una empresa o asociación.

```
In [17]: for i in range(len(df_eeuu[df_eeuu['sdnType'].isnull()])):
             df = df_eeuu[df_eeuu['sdnType'].isnull()].reset_index(drop=True)
             print(df.loc[i,'LastName'])
```

```
AEROCARIBBEAN AIRLINES
ANGLO-CARIBBEAN CO., LTD.
BANCO NACIONAL DE CUBA
BANCO NACIONAL DE CUBA
BANCO NACIONAL DE CUBA
BANCO NACIONAL DE CUBA
BOUTIQUE LA MAISON
CASA DE CUBA
CASA DE CUBA
CECOEX, S.A.
CIMEX
CIMEX IBERICA
CIMEX, S.A.
COMERCIAL IBEROAMERICANA, S.A.
COMERCIAL CIMEX, S.A.
COMERCIAL DE RODAJES Y MAQUINARIA, S.A.
COMERCIALIZACION DE PRODUCTOS VARIOS
COMPANIA DE IMPORTACION Y EXPORTACION IBERIA
CORPORACION CIMEX, S.A.
```

Figura 2: Bucle for para conocer los valores de la columna 'LastName' en los que 'sdnType' tiene valor nulo.

No obstante, existe un número minoritario de nombres en los que no queda claro qué tipo de sancionado puede ser. Se optará por pasar todos los valores nulos a valor 'enterprise'. Haciendo esto se introduce un cierto error en la clasificación de los registros por tipo de sancionado. Pero a cambio se recupera una gran cantidad de información que antes aparecía como nula.

La columna 'Address' también presenta gran cantidad de valores nulos. Como los nulos de esta columna no van a tener especial relevancia en los estudios posteriores, podemos dejarla como está.

Lista de la Unión Europea

Para las listas de la UE, la petición a la URL se hace con un token. No sirve el código utilizado con las listas de Estados Unidos porque no acepta el método wget.download(). Es necesario utilizar un proceso alternativo que no almacena la tabla en local, si no que genera directamente en Python un archivo con los datos [4].

Una vez se tiene este archivo, se le aplican diversas modificaciones para pasarlo a un DataFrame al que se llama *df_ue*. Cuando se tienen los datos en crudo de la lista de la UE como DataFrame, se pasan como .csv a la ubicación local *TFM/Código y datos/UE_EEUU_data_crudo.* Cada vez que se ejecuta el proceso, se elimina el .csv con los datos antiguos y se cargan los nuevos datos.

El DataFrame con los datos da la Unión Europea presenta muchas columnas. Por ello, el mejor método para identificar cuáles de estas son relevantes para nuestro estudio es dividir el DataFrame en grupos de 15 columnas y analizar separadamente las columnas de estas subtablas. La información sobre las variables seleccionadas tras realizar este análisis se encuentra en la Figura 3.

```
df_ue.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 21575 entries, 0 to 21574
Data columns (total 4 columns):
 #   Column                                Non-Null Count  Dtype 
---  ------                                --------------  ----- 
 0   Entity_SubjectType_ClassificationCode  20556 non-null  object
 1   NameAlias_WholeName                   14694 non-null  object
 2   Address_City                          5272 non-null   object
 3   Address_CountryDescription            4111 non-null   object
dtypes: object(4)
memory usage: 842.8+ KB
```

Figura 3: Información de metadatos de la tabla df_ue.

Los valores correspondientes a la ubicación ('Address_City' y 'Address_CountryDescription') no aparecen en el mismo registro que el nombre correspondiente a dicha ubicación. La información de nombres viene dada en la columna 'NameAlias_WholeName'. Para poder identificar los valores para la ubicación y moverlos al registro con su correspondiente nombre, es necesario agrupar por la columna 'Entity_EU_ReferenceNumber'. Esta columna engloba bajo el mismo identificador a un grupo de sancionados que comparten la misma ubicación.

Lo que se pretende agrupando por cada valor único 'Entity_EU_ReferenceNumber' es generar un nuevo DataFrame, llamado *df*, en el que a cada número de referencia se le asocie una ubicación descrita en términos de país y ciudad, tal y como se muestra en la Figura 4.

Una vez definido *df*, se hace join con *df_ue* para trasladar su información de la ubicación a la tabla de sancionados principal. Hecho esto se elimina la columna 'Entity_EU_ReferenceNumber', pues ya no es necesaria.

```
df.head(15)
```

	Entity_EU_ReferenceNumber	Address_City	Address_CountryDescription
0	EU.1001.60	0	0
1	EU.1003.87	0	0
2	EU.1006.79	0	0
3	EU.101.7	0	0
4	EU.1010.96	Tangerang	0
5	EU.1013.88	0	0
6	EU.1014.53	0	0
7	EU.1026.81	0	0
8	EU.103.9	0	0
9	EU.1033.90	0	0
10	EU.1034.55	0	Algeria
11	EU.1035.20	0	0
12	EU.1039.74	0	0
13	EU.1042.29	0	Italy
14	EU.1048.13	0	0

Figura 4: Tabla con la información de ubicación para cada número de referencia. Se observa que algunos de los sancionados de la UE no van a presentar información sobre su ubicación o va a estar incompleta.

En el momento en que fue realizado este análisis sobre la lista de la UE, la columna con la información sobre el tipo de entidad, 'Entity_SubjectType_ClassificationCode', presentaba 16 valores únicos contando el valor nulo. Son demasiados, ya que solo interesan rasgos más generales del tipo de sancionado como si es una empresa, una persona o una embarcación. Representamos 'Entity_SubjectType_ClassificationCode' en el gráfico de barras de la Figura 5:

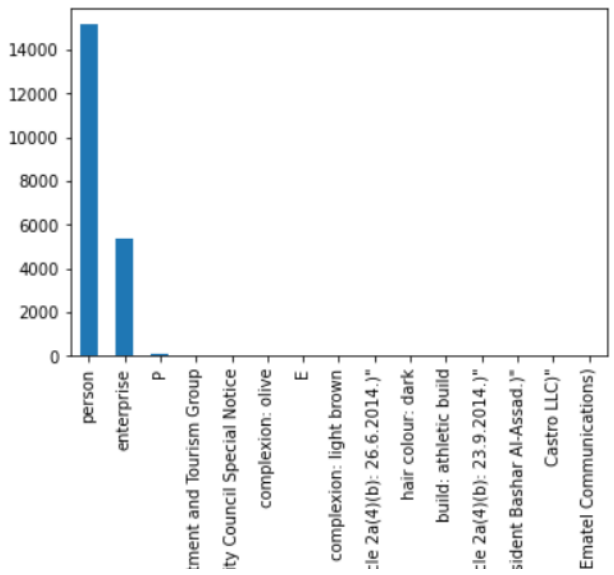

Figura 5: Gráfico de barras de la cardinalidad para cada valor de la columna 'Entity_SubjectType_ClassificationCode' de la tabla df_ue

Se aprecia que para casi todos los registros el valor no nulo de 'Entity_SubjectType_ClassificationCode' es 'person' o 'enterprise'. Por lo tanto, se pueden eliminar los otros registros sin que se pierda demasiada información.

Las categorías extra en 'Entity_SubjectType_ClassificationCode' aparecieron en una de las últimas versiones de la lista. Inicialmente, solo existían las categorías 'person' y 'enterprise'. El hecho de que aparezcan estas nuevas categorías puede deberse a un error humano o de software del lado del proveedor de datos.

Tras finalizar el proceso de extracción, los DataFrames *df_ue* y *df_eeuu* se llevan, en formato .csv, a la ubicación *TFM/Código y datos/datos_entrada_script_02.* Que es el fichero desde donde se toman los datos de entrada al script *02_transformar_datos.ipynb* para continuar con la depuración de las tablas.

2.1.2 Transformación y Limpieza de Datos

Para la transformación de los valores de las tablas obtenidas en el script 01 también se utiliza un notebook de Jupyter. Estas transformaciones buscan hacer los valores lo más adecuados posibles para su empleo en la herramienta de visualización y en los modelos de Machine Learning.

El script encargado de la transformación de los datos es *02_transformar_datos.ipynb.* Este script toma las tablas *df_ue.csv* y *df_eeuu.csv* en *TFM/Código y datos/datos_entrada_script_02* y las carga como DataFrame.

Los DataFrames resultantes se almacenan en formato .csv en *TFM/Código y datos/datos_entrada_script_03,* que es la ubicación desde la que accede el script 03 a los datos transformados.

Lista de Estados Unidos

Primera Transformación:

Para las primeras transformaciones sobre los valores de *df_eeuu* se crea un nuevo DataFrame llamado *df_eeuu_t1.* El 't1' en el nombre implica que esta es la primera transformación hecha sobre los datos tras ser cargados. Este DataFrame:

- Presenta una nueva variable llamada 'Origin' de valor 'EEUU' para identificar cual es la lista de origen.

- Se le han eliminado los registros con 'LastName' nulo. Ya que no podemos identificar un sancionado si no tiene un nombre.

- En caso de existir, se eliminan los registros duplicados. Para que se eliminen los registros duplicados pero que se diferencian por las mayúsculas y minúsculas, se ponen todas las variables en mayúsculas.

- Se cambia el nombre de algunas columnas para darles un nombre más conciso (Fig. 6). Este será el mismo nombre que se dé a las variables de la lista de la UE para que las filas puedan concatenarse sin problemas. Esta concatenación de las tablas se hará en el script 03, del que se hablará más adelante.

Segunda Transformación:

Se crea una nueva tabla llamada *df_eeuu_t2* para aplicar la segunda transformación sobre los datos: en algunos valores de la columna 'City' aparecen códigos y números que no interesan (Fig. 6), pues solo interesa el nombre de la ciudad. Se aplica una transformación mediante una expresión regular para eliminar estos códigos [5]. Ha sido necesario cambiar los valores perdidos en 'City' a cadena de caracteres para poder escanearlos por la expresión regular.

Una vez aplicada la expresión, se han vuelto a poner como valor 'Not a Number' (NaN). Se ha empleado el método re.sub() dentro de un bucle for que selecciona cada registro de la columna 'City' y le pasa la expresión regular. Los nuevos valores para 'City' se van almacenando en una lista con cada ejecución del bucle.

Finalmente, esta lista se sustituye por la columna 'City' antigua en *df_eeuu_t2.* Las pruebas para crear la expresión regular que eliminase las cadenas que no son de interés se han hecho en la página:

https://regex101.com/r/3vtVKW/1

En esta página se puede ver un cuadro con las cadenas de los valores de 'City'. Las subcadenas que han sido eliminadas aparecen resaltadas a color. La expresión regular que selecciona estas subcadenas aparece encima del cuadro.

```
df_eeuu_t1
```

	Name	EntityType	Address	City	Country	Origin
0	AEROCARIBBEAN AIRLINES	ENTERPRISE	NaN	HAVANA	CUBA	EEUU
1	ANGLO-CARIBBEAN CO., LTD.	ENTERPRISE	IBEX HOUSE, THE MINORIES	LONDON EC3N 1DY	UNITED KINGDOM	EEUU
2	BANCO NACIONAL DE CUBA	ENTERPRISE	ZWEIERSTRASSE 35	ZURICH CH-8022	SWITZERLAND	EEUU
3	BANCO NACIONAL DE CUBA	ENTERPRISE	AVENIDA DE CONCHA ESPINA 8	MADRID E-28036	SPAIN	EEUU
4	BANCO NACIONAL DE CUBA	ENTERPRISE	DAI-ICHI BLDG. 6TH FLOOR, 10-2 NIHOMBASHI, 2-C...	TOKYO 103	JAPAN	EEUU
...	...	...	...	...	...	...
15282	PIONEER SHIPMANAGEMENT PTE. LTD.	ENTERPRISE	24-07, SHENTON HOUSE, 3, SHENTON WAY	SINGAPORE 068805	SINGAPORE	EEUU
15283	GOLDEN WARRIOR SHIPPING CO. LIMITED	ENTERPRISE	9TH FLOOR, BLOCK C, QINGDAO PLAZA, 381, DUNHUA...	QINGDAO, SHANDONG 266034	CHINA	EEUU
15284	GOLDEN WARRIOR SHIPPING CO. LIMITED	ENTERPRISE	UNIT D, 16/F, ONE CAPITAL PLACE, 18 LUARD ROAD...	HONG KONG	CHINA	EEUU
15285	GLORY HARVEST	VESSEL	NaN	NaN	NaN	EEUU
15286	TORNADO CASH	ENTERPRISE	NaN	NaN	NaN	EEUU

15280 rows × 6 columns

Figura 6: Tabla de Estados Unidos tras aplicar la primera transformación. Se aprecia que algunos valores de la columna 'City' presentan códigos tras el nombre de la ciudad y que se han cambiado los nombres de columna respecto a la tabla sin transformar.

	Name	EntityType	Address	Country	Origin	City
0	AEROCARIBBEAN AIRLINES	ENTERPRISE	NaN	CUBA	EEUU	HAVANA
1	ANGLO-CARIBBEAN CO., LTD.	ENTERPRISE	IBEX HOUSE, THE MINORIES	UNITED KINGDOM	EEUU	LONDON
2	BANCO NACIONAL DE CUBA	ENTERPRISE	ZWEIERSTRASSE 35	SWITZERLAND	EEUU	ZURICH
3	BANCO NACIONAL DE CUBA	ENTERPRISE	AVENIDA DE CONCHA ESPINA 8	SPAIN	EEUU	MADRID
4	BANCO NACIONAL DE CUBA	ENTERPRISE	DAI-ICHI BLDG. 6TH FLOOR, 10-2 NIHOMBASHI, 2-C...	JAPAN	EEUU	TOKYO
...	...	...	...	...	...	...
15260	GLORY HARVEST	VESSEL	NaN	NaN	EEUU	NaN
15261	TORNADO CASH	ENTERPRISE	NaN	NaN	EEUU	NaN
15262	TWEHWAY, BILL	INDIVIDUAL	NaN	LIBERIA	EEUU	MONROVIA
15263	CEPHUS, SAYMA SYRENIUS	INDIVIDUAL	NaN	LIBERIA	EEUU	NaN
15264	MCGILL, NATHANIEL	INDIVIDUAL	NaN	LIBERIA	EEUU	PAYNESVILLE

Figura 7: Tabla tras aplicar la segunda transformación. Su nombre es df_eeuu_t2

Teniendo en cuenta que los datos se van actualizando y cambian periódicamente, puede que la expresión regular aplicada en la segunda transformación a la columna 'City' no siempre aplique bien sobre todos los valores. Esto implica que nos podemos encontrar con una pequeña cantidad de valores para City con una transformación errónea por la expresión regular.

Tercera Transformación:

Revisando los datos se descubre que la expresión regular aplicada no ha conseguido eliminar algunos de los códigos que aparecen en las variables 'City'. Estos códigos son valores de códigos postales que no van acompañados del nombre de ninguna ciudad.

Se crea una tercera transformación sobre la tabla de Estados Unidos para eliminar estos valores. A la tabla con los valores de la tercera transformación se la llama *df_eeuu_t3*.

Se utilizará de nuevo una expresión regular; pero esta vez lo que hace es sustituir los valores numéricos de la columna 'City' por 'Not a Number'.

Finalmente se carga la tabla transformada final *df_eeuu_t3* en *TFM/Código y datos/datos_entrada_script_03.*

Lista de la Unión Europea

Primera Transformación:

Las primeras transformaciones sobre los valores de *df_eu* siguen una estructura similar a las de la tabla de Estados Unidos. Se crea un DataFrame llamado *df_eu_t1* sobre el que se aplican las siguientes transformaciones:

- Se añade una nueva variable llamada 'Origin' de valor 'UE' para identificar cual es la lista de origen.

- Se han eliminado los registros con 'NameAlias_WholeName' nulo. Ya que no podemos identificar un sancionado si no tiene un nombre.

- Se eliminan los registros duplicados. Para que se eliminen los registros duplicados pero que se diferencian por las mayúsculas y minúsculas, se ponen todas las variables en mayúsculas.

- Se cambia el nombre de las columnas para darles un nombre más conciso como se muestra en la Figura 8. Este será el mismo nombre que se ha dado a las variables de la lista de EEUU para que puedan concatenarse sin problemas en el script 03.

Echando un vistazo a los valores de la columna 'Name' de *df_eu_t1* (Fig. 8), Se observan dos problemas con sus valores:

1. Aparecen nombres en la columna 'Name' en idiomas que utilizan alfabetos distintos al inglés o al español, dificultando su comprensión. Sería útil poder traducir, en la medida de lo posible, estos nombres al inglés para que sean más comprensibles.

2. Hay registros muy parecidos salvo por pequeñas diferencias en los valores en la columna 'Name' que pueden considerarse duplicados. Esto puede deberse a que el sancionado escribió mal su nombre en algún formulario del que luego se pasó esta información a la lista de sancionados, o al uso de un alias por parte del sancionado. También puede ocurrir que, aunque los nombres se parezcan mucho, correspondan a distintos sancionados.

   En cualquier caso, como este trabajo no busca el seguimiento individual de cada sancionado, si no una visión más general de los tipos de sancionados y sus ubicaciones, se considerarán los nombres similares que tengan el resto de variables iguales como duplicados.

```
df_ue_t1
```

	EntityType	Name	City	Country	Origin
0	PERSON	SADDAM HUSSEIN AL-TIKRITI	NaN	NaN	UE
1	PERSON	ABU ALI	NaN	NaN	UE
2	PERSON	ABOU ALI	NaN	NaN	UE
3	PERSON	QUSAY SADDAM HUSSEIN AL-TIKRITI	NaN	NaN	UE
4	PERSON	QOUSSAÏ SADDAM HUSSEIN AL-TIKRITI	NaN	NaN	UE
...	...	...	...	...	...
14530	PERSON	ЄВГЕН ВІТАЛІЙОВИЧ БАЛИЦЬКИЙ	NaN	NaN	UE
14531	PERSON	YEVHEN VITALIIOVYCH BALYTSKIY	NaN	NaN	UE
14532	PERSON	ЕВГЕНИЙ ВИТАЛЬЕВИЧ БАЛИЦКИЙ	NaN	NaN	UE
14533	PERSON	YEVGENIY VITALIEVICH BALYTSKIY	NaN	NaN	UE
14534	PERSON	JEVGENIJ VITALJEVITJ BALITSKIJ	NaN	NaN	UE

14535 rows × 5 columns

Figura 8: Tabla de la Unión Europea tras aplicar la primera transformación. En la columna 'Name' aparecen nombres en distintos idiomas y algunos de ellos muy similares entre sí. Se han cambiado los nombres de columna respecto a la tabla sin transformar.

Segunda Transformación:

Para resolver la problemática del punto 1 se han probado varios módulos de Python, como goslate y googletrans, que hacen una petición a un servicio de traducción en línea. Ambos módulos daban problemas relacionados con la versión y el número límite de peticiones que se podían hacer. Se terminó por utilizar el módulo de deepl. Esta librería de Python da acceso a la API de DeepL. DeepL es un traductor en línea que utiliza tecnología basada en Inteligencia Artificial para obtener mejores resultados. Para poder utilizar DeepL desde Python, es necesario crear una cuenta gratuita que genere una API key que permita la autentificación del cliente. El registro de cuenta se hace desde la web oficial de la API DeepL:

https://www.deepl.com/docs-api

Se pasó un bucle for a los registros de la columna 'Name' para traducirlos y se llevó cada nombre traducido a una lista. Finalmente se sustituyeron los valores en la columna 'Name' en el DataFrame *df_ue_t1* por los valores de la lista y se renombró como *df_ue_t2* (ver Fig. 9). Algunos de los valores, como los nombres en árabe, no pudieron ser traducidos. No obstante, el resultado general de la traducción es bueno.

	Entity Type	Name	City	Country	Origin
0	PERSON	SADDAM HUSSEIN AL-TIKRITI	NaN	NaN	UE
1	PERSON	ABU ALI	NaN	NaN	UE
2	PERSON	ABOU ALI	NaN	NaN	UE
3	PERSON	QUSAY SADDAM HUSSEIN AL-TIKRITI	NaN	NaN	UE
4	PERSON	QOUSSAÏ SADDAM HUSSEIN AL-TIKRITI	NaN	NaN	UE
...	...	...	...	...	...
14530	PERSON	YEVGENY VITALIOVICH BALITSKY	NaN	NaN	UE
14531	PERSON	YEVHEN VITALIIOVYCH BALYTSKIY	NaN	NaN	UE
14532	PERSON	EVGENY VITALIEVICH BALITSKY	NaN	NaN	UE
14533	PERSON	YEVGENIY VITALIEVICH BALYTSKIY	NaN	NaN	UE
14534	PERSON	YEVGENY VITALIEVICH BALITSKY	NaN	NaN	UE

Figura 9: Tabla de la Unión Europea tras aplicar la segunda transformación. En la columna 'Name' se puede ver cómo los últimos registros han sido traducidos.

Tercera Transformación:

Todavía existe el problema expuesto en el punto 2: existen nombres escritos de forma muy similar que podrían ser duplicados. Para solucionarlo se crea una tercera transformación sobre el DataFrame y se renombra como *df_ue_t3*. Se emplea la función 'get_close_matches()' del módulo de Python llamado difflib [6].

La función se emplea dentro de un bucle for que comprueba para cada registro cuantos registros similares o iguales tiene debajo. Si se detecta que no hay ningún registro similar, este es mandado a la lista 'new_list_name' con un condicional de tipo if. Cada elemento en 'new_list_name' representa un registro con las columnas separadas por guiones. La función presenta dos parámetros ajustables:

- *n* tiene valor 4 por defecto e indica el número máximo de registros que, definido un cierto nombre, pueden ser similares a este. Es decir, para cada nombre el número máximo de nombres que pueden considerarse duplicados debido a su similitud y por tanto pueden ser eliminados es de 4 nombres.

- *sensibility* es un número entre 0 y 1 que indica el grado de sensibilidad con el que funciona el detector de matches. A mayor sensibilidad, más registros serán considerados como duplicados y, en consecuencia, menos registros aparecen en la lista final. Se hicieron varias pruebas ajustando la sensibilidad hasta encontrar el valor que eliminase la mayor cantidad de registros similares sin que se perdiera excesiva información.

	Name	City	Country	EntityType	Origin
0	SADDAM HUSSEIN AL-TIKRITI	NaN	NaN	PERSON	UE
1	ABOU ALI	NaN	NaN	PERSON	UE
2	QOUSSAÏ SADDAM HUSSEIN AL-TIKRITI	NaN	NaN	PERSON	UE
3	UDAY SADDAM HUSSEIN AL-TIKRITI	NaN	NaN	PERSON	UE
4	ELDERSA SADDAM HUSSEIN AL-TIKRITI	NaN	NaN	PERSON	UE
...	...	...	...	...	...
10564	VIKTOR YANUKOVYCH	NaN	NaN	PERSON	UE
10565	VIKTOR FEDOROVICH YANUKOVYCH	NaN	NaN	PERSON	UE
10566	YEVHEN VITALIIOVYCH BALYTSKIY	NaN	NaN	PERSON	UE
10567	YEVGENIY VITALIEVICH BALYTSKIY	NaN	NaN	PERSON	UE
10568	YEVGENY VITALIEVICH BALITSKY	NaN	NaN	PERSON	UE

Figura 10: Tabla transformada final df_ue_t3

Finalmente se carga la tabla transformada final *df_ue_t3 (Fig. 10)* en *TFM/Código y datos/datos_entrada_script_03.*

Se aprecia que el resultado final de las transformaciones es más completo para la tabla de Estados Unidos que para la tabla de la Unión Europea. La tabla de la UE presenta más valores nulos y no presenta la columna 'Address' que sí aparece en la tabla de EEUU.

2.1.3 Integración de Datos

Para terminar con los procesos manipulación de los datos, se cargan las tablas obtenidas tras ejecutar el script *02_transformar_datos.ipynb* en *03_concatenar_datos.ipynb.* Este script simplemente unirá en una misma tabla, con la función concat() de la libraría pandas, las tablas de sancionados de la UE y de EEUU; como puede verse en la Figura 11.

De este modo, se tiene toda la información centralizada en un único dataset depurado que puede cargarse en Power BI o en cualquier otra herramienta de análisis de datos que se quiera emplear.

Se pueden analizar los valores de la tabla haciendo uso de funciones de pandas como info(), describe() o unique(). Se encuentra lo siguiente:

- Existen nombres repetidos. Estos nombres se corresponden a sancionados presentes en varias ubicaciones. Por ejemplo, una empresa que desarrolle actividad comercial en varios países.

- Los sancionados de tipo 'PERSON' e 'INDIVIDUAL' realmente son el mismo tipo de sancionado. Se cambia el valor 'INDIVIDUAL' por 'PERSON' en todos los registros.

```
sanctionlist = pd.concat([df_eeuu, df_ue]).reset_index(drop=True)
```

```
sanctionlist
```

	Name	EntityType	Address	Country	Origin	City
0	AEROCARIBBEAN AIRLINES	ENTERPRISE	NaN	CUBA	EEUU	HAVANA
1	ANGLO-CARIBBEAN CO., LTD.	ENTERPRISE	IBEX HOUSE, THE MINORIES	UNITED KINGDOM	EEUU	LONDON
2	BANCO NACIONAL DE CUBA	ENTERPRISE	ZWEIERSTRASSE 35	SWITZERLAND	EEUU	ZURICH
3	BANCO NACIONAL DE CUBA	ENTERPRISE	AVENIDA DE CONCHA ESPINA 8	SPAIN	EEUU	MADRID
4	BANCO NACIONAL DE CUBA	ENTERPRISE	DAI-ICHI BLDG. 6TH FLOOR, 10-2 NIHOMBASHI, 2-C...	JAPAN	EEUU	TOKYO
...	...	...	...	...	...	...
25822	VIKTOR YANUKOVYCH	PERSON	NaN	NaN	UE	NaN
25823	VIKTOR FEDOROVICH YANUKOVYCH	PERSON	NaN	NaN	UE	NaN
25824	YEVHEN VITALIIOVYCH BALYTSKIY	PERSON	NaN	NaN	UE	NaN
25825	YEVGENIY VITALIEVICH BALYTSKIY	PERSON	NaN	NaN	UE	NaN
25826	YEVGENY VITALIEVICH BALITSKY	PERSON	NaN	NaN	UE	NaN

25827 rows × 6 columns

Figura 11: Tabla final de sancionados

- Hay muchos valores nulos para la columna 'Address' porque no se tiene información de la dirección para la lista de la UE. Únicamente se conoce la dirección para algunos registros de la lista de EEUU.

- Hay bastante nulos para el país y la ciudad. La mayoría procedentes de la tabla de la Unión Europea.

Finalmente, se lleva el DataFrame concatenado a la carpeta 'datos_entrada_script_04_y_PowerBI' desde donde cargado en Power BI para la generación del dashboard.

2.2 Modelos de Aprendizaje No Supervisado

Se intentó usar K-Medias para agrupar los registros y PCA para reducir la dimensionalidad de las variables. Como métrica para evaluar la calidad del modelo se calculó el coeficiente de silueta.

A la vista de los resultados y de la falta de variables de calidad para entrenar los modelos se descartó la aplicación de métodos de aprendizaje no supervisado. La razón de esto es que las variables eran categóricas nominales y no numéricas, además de que algunas de ellas presentan una cardinalidad muy alta.

Por todo lo anterior, además de que la evaluación no dio resultados muy buenos, se descartó el uso de algoritmos de Machine Learning para estudiar los datos.

A continuación, se describe el funcionamiento de los algoritmos que fueron necesarios en la construcción del modelo de aprendizaje no supervisado y que finalmente no fueron aplicados.

Codificación

Todas las variables son categóricas nominales. Hay que sustituirlas por valores numéricos por algún método de codificación. Se emplean tres métodos de codificación. La elección de uno u otro dependerá del tipo de variable y de lo que se busque conseguir con el modelo:

- **BinaryEncoder**: Es una clase de la librería category_encoders de Python. Su funcionamiento se puede dividir en los siguientes pasos [13]:

  - Cada categoría única de la variable categórica seleccionada se convierte en un número aleatorio desde el 1 hasta el número total de categorías únicas de la variable.

  - A continuación, estos valores se transforman a código binario.

  - Por último, cada dígito binario se separa en distintas columnas, una por cada dígito.

La siguiente formula, cuyo resultado deberemos redondear hacia arriba [13], muestra el número de columnas que se formarán al aplicar BinaryEncoder:

$$\frac{\log(n+1)}{\log(2)}$$

donde n es el número total de categorías.

- **OneHotEncoder**: Pertenece al paquete sklearn.preprocessing de la librería scikit-learn de Python. La estrategia que implementa es crear una columna para cada valor distinto que exista en la característica que estamos codificando y, para cada registro, marcar con un 1 la columna a la que pertenezca dicho registro y dejar las demás con 0 [14].

  El inconveniente de esta técnica frente a BinaryEncoder es que crea muchas más columnas. El hecho de usar muchas columnas para entrenar un modelo de Machine Learning puede llevar a problemas de rendimiento y de calidad del modelo.

- **OrdinalEncoder:** También pertenece al paquete sklearn.preprocessing**.** Simplemente asigna un valor entero a cada valor categórico en función de su orden alfabético.

  Como ya se ha comentado, todas las variables son categóricas nominales. Es decir, no siguen un orden. El problema de utilizar un codificador ordinal es que jerarquiza los

valores según su orden alfabético, pudiendo entrañar problemas en la interpretación de los datos por parte del modelo.

Imputación

La imputación consiste en la sustitución de los valores nulos del dataset de entrada por otros siguiendo un cierto criterio. Los tipos de imputadores utilizados fueron los siguientes:

- **SimpleImputer:** Pertenece al paquete sklearn.impute de scikit-learn en Python. Sustituye el valor de los nulos por un cierto valor según la estrategia seleccionada. El parámetro *strategy* permite seleccionar hasta 4 tipos distintos de sustituciones [15]:
  - 'mean' reemplaza los valores nulos por la media para cada columna.
  - 'median' reemplaza los valores nulos por la mediana para cada columna.
  - 'most_frequent' sustituye el valor nulo utilizando el valor más frecuente para cada columna.
  - 'constant' sustituye el valor nulo por un cierto valor definido por el parámetro *fill_value*.

- **KNNImputer**: También pertenece al paquete sklearn.impute. Cada registro con valores nulos se imputa usando el valor medio de los vecinos cercanos. El parámetro *n_neighbors* permite seleccionar el número de vecinos cercanos.

Reducción de la dimensionalidad

El análisis de componentes principales (PCA) es, con diferencia, el algoritmo de reducción de dimensionalidad más popular. En primer lugar, identifica el hiperplano que queda más cerca de los datos y, a continuación, proyecta en él los datos.

El hiperplano más adecuado será el que preserva la máxima cantidad de varianza, ya que lo más probable es que sea el que pierde menos información. Otra manera de justificar esta elección es que se trata del hiperplano que minimiza la distancia cuadrática media entre el conjunto de datos original y su proyección en ese hiperplano [11].

Para los modelos desarrollados se utilizan dos variantes de PCA:

- **PCA:** Pertenece al paquete sklearn.decomposition. El parámetro ajustable *n_clusters* permite seleccionar el número de dimensiones con las que se queda el algoritmo. Con el método *explained_variance_ratio_* se conoce la proporción de la varianza del conjunto de datos que queda a lo largo de cada dimensión seleccionada por PCA [11].

Clusterización

Se utilizó la clase **KMeans** del paquete sklearn.cluster para agrupar los registros. Su funcionamiento es el siguiente:

1. Se generan de forma aleatoria unos puntos llamados centroides. La cantidad de centroides dependerá del valor pasado al parámetro *n_clusters.*

2. Se etiquetan los registros en función del centroide que tengan más cerca para crear los grupos.

3. Con las instancias etiquetadas, se actualiza la posición de los centroides a partir de la media de las instancias para cada grupo.

4. Con los centroides actualizados, se pueden etiquetar de nuevo las instancias.

5. Se repite el proceso de los puntos 3 y 4 hasta que el algoritmo converge cuando los centroides dejan de moverse.

K-Means necesita como parámetro de entrada el número de grupos *n_clusters*. Para conocer el número óptimo de grupos, se utiliza el **coeficiente de silueta.** Para una instancia, este coeficiente es igual a *(b – a) / máx(a,b)* [11], donde *a* es la distancia media a las otras instancias en el mismo grupo y *b* es la distancia media al grupo más cercano.

El coeficiente de silueta varía entre -1 y +1. Cuanto más cercano a +1, más metida está la instancia en su propio grupo y lejos del resto; mientras que un coeficiente cercano a 0 significa que la instancia se sitúa cerca del límite del grupo al que pertenece. Por último, un coeficiente negativo y cercano a -1 supone que la instancia podría no estar bien clasificada en el grupo al que pertenece.

Se utilizó la función silhouette_score() para conocer el coeficiente medio de silueta para cada número de grupos seleccionado.

2.3 Generación de Dashboards

A partir de los datos depurados siguiendo los pasos expuestos en la sección 2.1, se generan dos cuadros de mando utilizando Power BI. El primero será un cuadro de mando más general en el que la información girará principalmente en torno al nombre del sancionado. El segundo seleccionará aquellos sancionados de los que se conoce la ubicación para extraer información acerca de este grupo específico de sancionados.

Un punto interesante al manejar Power BI fue que permitió descubrir algunos fallos en los datos que habían pasado desapercibidos al momento de su transformación y limpieza.

Al tratarse de una herramienta de visualización, permite una mejor visión general de los datos que Python. Los errores descubiertos fueron:

- Se descubre gracias a Power BI que existían valores nulos para la columna 'Country' etiquetados como 'UNKNOWN'. Se debe modificar para cambiarlo a NaN como el resto de los valores nulos.

- Revisando los datos con Power BI se descubre que la expresión regular aplicada no ha conseguido eliminar algunos de los códigos que aparecen en la variable 'City'. Estos códigos son valores de códigos postales que no van acompañados del nombre de ninguna ciudad. Estos códigos deben ser sustituidos por NaN.

2.3.1 Preprocesamiento

En primer lugar, desde la pestaña de Datos de Power BI se realizan una serie de cambios a nivel de metadatos y sobre los valores:

- Se clasifican las columnas ‘Country’, ‘City’ y ‘Address’ como columnas de país o región, ciudad y dirección, respectivamente. Esto para facilitar a Power BI la tarea de representar ubicaciones a partir de estos datos.

- Los valores que eran de tipo nulo en el formato de entrada de los datos son puestos como una celda vacía en Power BI. Se cambia a valor ‘UNKNOWN’ para no tener valores vacíos en las representaciones.

- Se crea con código DAX una nueva columna llamada ‘condicion_cm_ubicaciones’ que devuelve *True* si la fila tiene país y ciudad desconocidos y *False* en caso contrario. Esto se hace para aplicar un filtro a toda la página del cuadro de mando de ubicaciones para que solo represente datos de los que se conozca la ubicación.

A continuación, se especifican los objetos utilizados en cada cuadro de mando y cómo son las interacciones entre los mismos. Cada cuadro de mando se muestra en una página independiente.

2.3.2 Cuadro de Mando General de Sancionados

❖ **Cuadro general (ver Fig. 14)**: Los objetos utilizados en la página de Power BI que contiene el cuadro de mando general son los siguientes:

- **Tarjeta (1)** donde se muestra el nombre de lista de origen con más registros para los datos seleccionados. En el caso de que se seleccione un solo dato

muestra el origen de este. Puede tomar dos valores: EEUU (Estados Unidos) y UE (Unión Europea).

- **Gráfico de anillos (2)** con la información del recuento de cada tipo de sancionado para los datos seleccionados acompañada de una leyenda. Si solo se selecciona un dato, todo el anillo cambia al color reservado para ese tipo de sancionado.

- **Text Filter (3)** es un buscador que filtra en el resto de los objetos por nombre.

- **Tabla (4).** Tabla con la información completa de nombre, país, ciudad y dirección para los datos seleccionados.

- **Treemap (5)** que muestra aquellas entidades que han sido sancionadas en más de 15 ubicaciones distintas. El número de ubicaciones en las que han sido sancionadas aparece abajo a la izquierda de cada panel.

  Este objeto está vinculado con una página llamada **Tooltip.** Cuando se pasa el ratón sobre uno de los paneles, aparece el nombre completo del sancionado procedente de una tabla generada en la página Tooltip. El detalle de este tooltip aparece en la Figura 15.

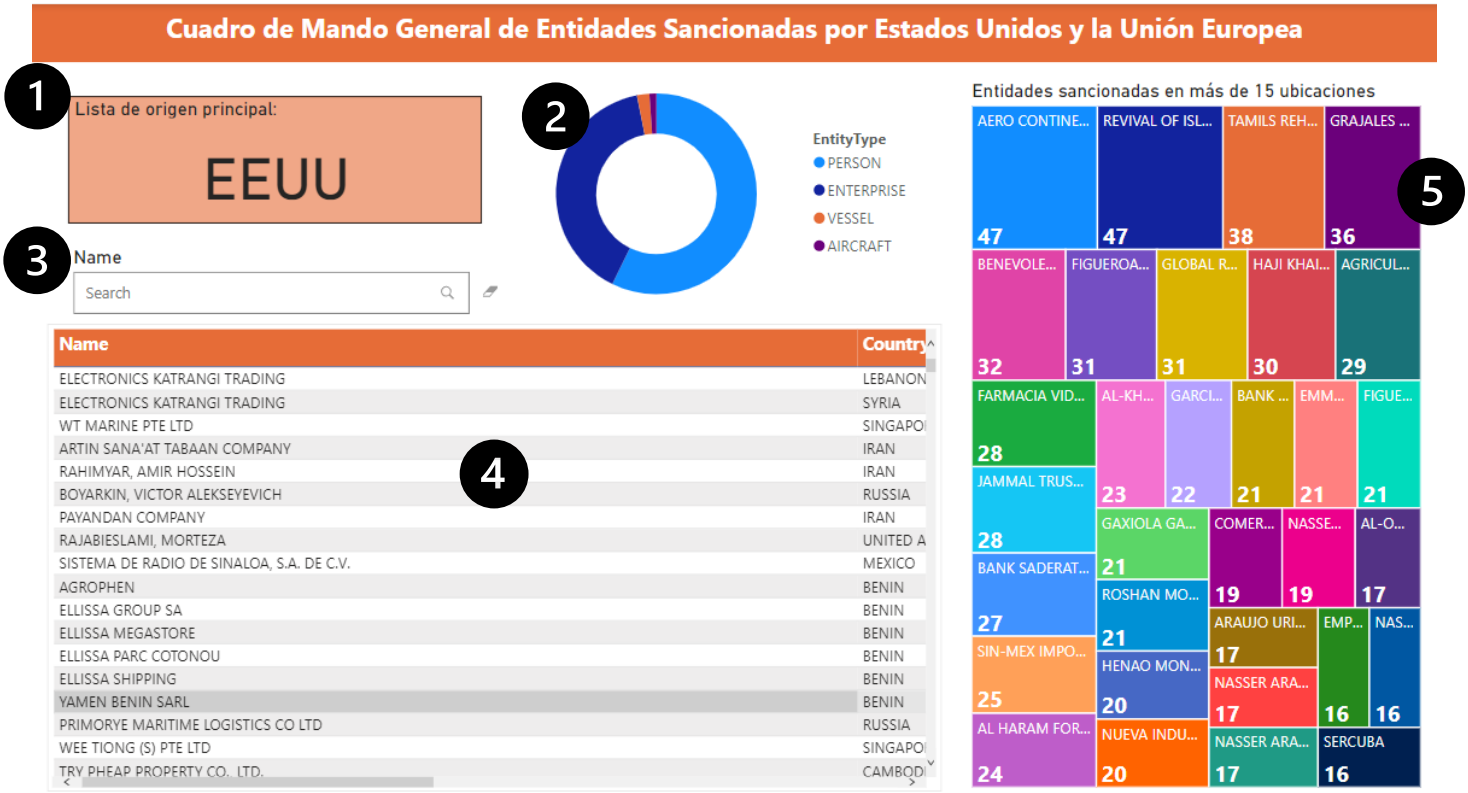

Figura 14: Cuadro de mando general. Los números en círculos negros relacionan cada objeto del cuadro con su descripción.

Todos estos objetos se encuentran interrelacionados. Esto es, las selecciones hechas en uno de ellos afectan al resto y viceversa.

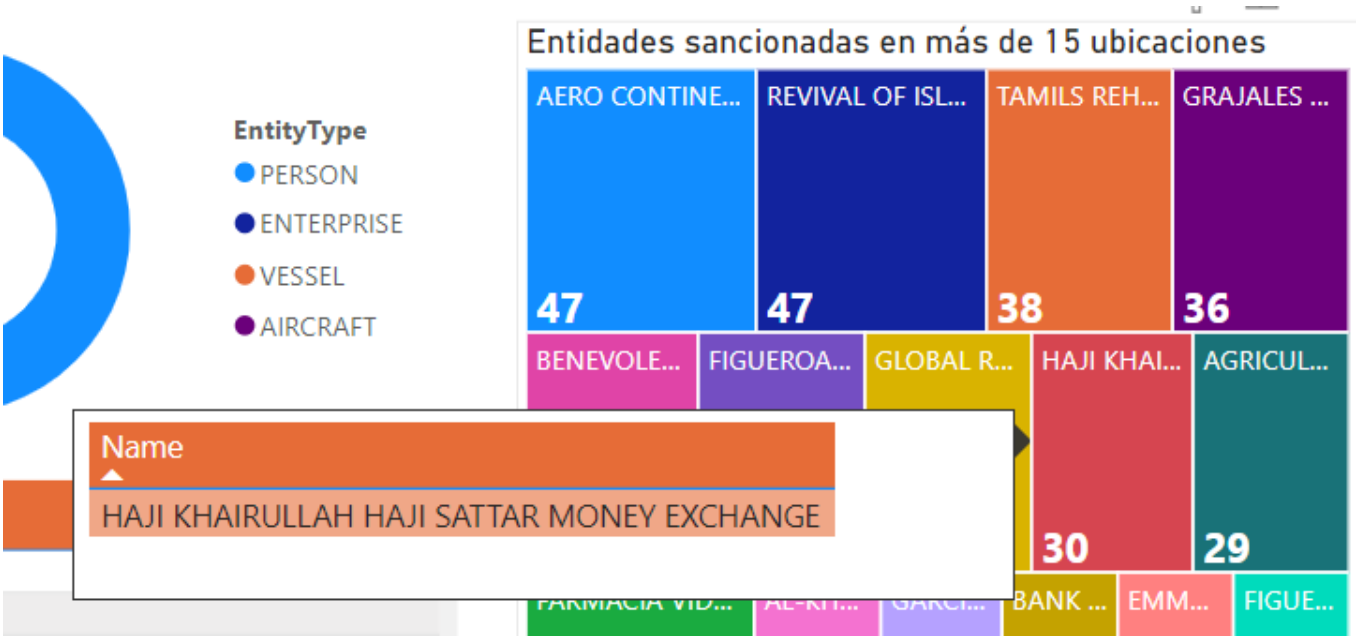

Figura 15: Detalle del cuadro de mando general donde se ve como aparece el tooltip creado con el nombre completo. Se activa al pasar el ratón sobre un panel del treemap.

2.3.3 Cuadro de Mando de Ubicaciones de Sancionados

- **Cuadro de ubicaciones (ver Fig. 16):** Los objetos utilizados en la página de PowerBI que contiene el cuadro de mando de ubicaciones son los siguientes:

  - **Mapa (1):** Representa con burbujas los sancionados sobre un mapa. El tamaño de la burbuja depende del número de sancionados en esa ubicación y el color de la lista de origen de los datos.

    Algunas de las burbujas están divididas en dos porciones de los dos colores posibles dependiendo del origen. Esto ocurre cuando para una misma ubicación hay sancionados tanto de la lista de Estados Unidos como de la Unión Europea.

  - **Forma (2):** Se trata de un botón para mostrar información más detallada.

    Cuando se hace click sobre una de las burbujas del mapa el botón cambia a color verde. Si en ese momento se hace click sobre el botón verde, nos lleva a la página llamada **Detalle,** que se muestra en la Figura 17, con información más detallada del grupo de sancionados pertenecientes a esa burbuja.

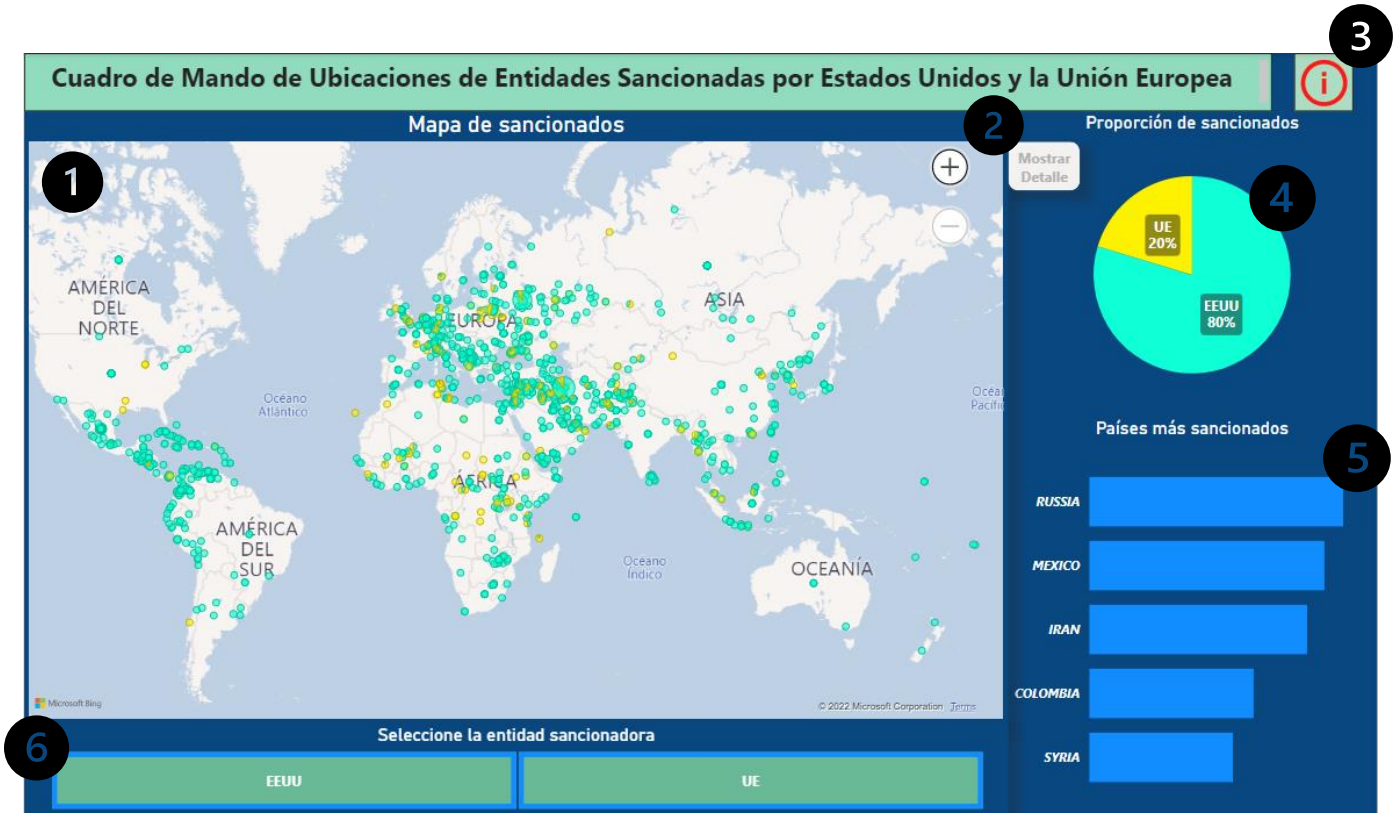

Figura 16: Cuadro de mando de ubicaciones. Los números en círculos negros relacionan cada objeto del cuadro con su descripción.

Figura 17: Página de detalle para el cuadro de mando de ubicaciones.

- **Botón (3):** Al hacer click sobre el botón se muestra un mensaje. El mensaje advierte sobre el uso únicamente de datos en los que se conoce la ubicación para este cuadro de mando. El detalle de esta advertencia se puede ver en la Figura 18.

- **Gráfico circular (4):** Muestra la proporción de cada origen de los datos para el conjunto de datos seleccionados.

- **Gráfico de barras (5):** Muestra los 5 países más sancionados. Estos cambiarán según la lista de origen seleccionada.

- **Segmentación de datos (6):** Doble botón que permite filtrar en función de su lista de origen. Puede seleccionarse Estados Unidos, la Unión Europea o ambas.

Estos objetos se encuentran interrelacionados. Esto es, las selecciones hechas en uno de ellos afectan al resto.

Como excepción, se ha eliminado la interacción al hacer click sobre una burbuja del mapa con el gráfico circular y el gráfico de barras porque no aportaba información de interés.

Figura 18: Detalle del cuadro de mando de ubicaciones donde se muestra el mensaje que aparece tras accionar el botón de información. Para devolver el cuadro a su estado original basta con hacer click sobre cualquier zona fuera del cuadro de texto.

Capítulo 3. CONCLUSIONES Y FUTURAS LÍNEAS DE TRABAJO

El funcionamiento de los cuadros de mando obtenidos tras el largo proceso es el esperado. De ellos se puede deducir gran cantidad de información que sería imposible conocer únicamente mirando las tablas de datos.

La información de estos cuadros podría ser útil, por ejemplo, para una empresa que quiere identificar entre sus clientes a personas a las que no deberían ofrecer sus servicios. Para ello, la empresa podría vincular la tabla final creada en este proyecto con las tablas de cliente de su base de datos y buscar posibles coincidencias.

Este trabajo me ha permitido poner en práctica mis habilidades en procesamiento de datos y aplicación de modelos de Machine Learning utilizando Python.

Desgraciadamente, los modelos de aprendizaje no supervisado no fueron lo suficientemente buenos y claros como para incluirlos en este trabajo. No obstante, intentar aplicar estos modelos me ha servido para poner en práctica los conocimientos que ya tenía en el campo del aprendizaje no supervisado y adquirir algunos nuevos.

El problema principal para poder aplicar los modelos de Machine Learning es que todas las variables del dataset eran categóricas y varias de ellas de alta cardinalidad. Este tipo de variables no son adecuadas para entrenar un modelo ML, ya que un modelo de estas características necesita valores numéricos que no sean identificadores únicos.

En cuanto a futuras líneas de trabajo, cabe destacar:

- La traducción y el fragmento de código que elimina los registros con nombres similares son muy lentos. Se planteó utilizar Spark pero habría que haber adaptado la lógica del código al de la aplicación de Big Data.

- Haber tenido variables numéricas habría haber permitido aplicar el modelo ML. Las variables numéricas podrían generarse, por ejemplo, a partir de la latitud y longitud de la ubicación.

- En Power BI se podría haber hecho más por tener una lista de los sancionados que aparezcan en ambas listas. De esta manera, si de uno no se conoce la ubicación en una lista, se puede deducir de su ubicación en la otra.

PRESUPUESTO Y TIEMPO

Las herramientas de software empleadas en este trabajo son open source. Para generar el código, tanto el lenguaje de programación (Python) como el entorno de desarrollo integrado o IDE (Jupyter Notebook) están disponibles a través de la web de forma gratuita. La herramienta de visualización e inteligencia empresarial empleada (Power BI) ha sido utilizada en su versión gratuita.

También se hizo uso de un traductor en línea (DeepL) para el cual el acceso a la API tenía una opción gratuita. Esta opción para el traductor solo permite ejecutar una traducción de la tabla al mes antes de que se agoten la cantidad de caracteres mensuales. De modo que si se quiere ejecutar más de una traducción al mes hay que pagar aproximadamente 7.50 € por cada nueva ejecución en el mismo mes. La cantidad exacta que pagar dependerá de la cantidad de sancionados que haya en la tabla en el momento de la traducción. Como es un servicio bajo demanda, a más sancionados habrá más cantidad de caracteres y el precio será mayor.

Hay que tener en cuenta la inversión inicial en un ordenador con la capacidad de cómputo necesaria y un sistema operativo que permita la instalación de las últimas versiones de las aplicaciones utilizadas. También es necesario el acceso a internet para poder actualizar las tablas periódicamente.

	Descripción	Presupuesto	Opcional
Gastos de Capital (CAPEX)	Ordenador personal	250 – 1100 €	No
Gastos de Operación (OPEX)	Suscripción DeepL API Pro	4.99 – 15 €/mes	Sí
	Internet	31 – 33 €/mes	No

DIAGRAMA DE GANTT

Para entender la planificación del trabajo y ver el tiempo dedicado a cada tarea principal se hace un diagrama de Gantt. El eje vertical representa las tareas realizadas y el horizontal los meses y días:

Tarea	Julio	Agosto	Septiembre
Extraer y limpiar datos	10 días	12 días	
Redactar introducción	1 día	2 días	
Redactar memoria técnica	7 días	7 días	5 días
Entrenar modelos no supervisados		10 días	5 días
Redactar conclusiones			2 días
Hacer los dashboard			5 días

Se ve como el mayor tiempo ha sido dedicado a la depuración y procesamiento de datos en Python seguido de la elaboración de la memoria técnica. En total han sido unos 66 días de trabajo. A 3 horas por día, son unas **198 horas de trabajo**.

Como se ha comentado en las conclusiones, el entrenamiento de los modelos no supervisados no fue incluido en el trabajo porque no se llegó a tener buenos resultados.

BIBLIOGRAFÍA

[1] "¿Qué son las sanciones internacionales?" [Online]. Available: https://elordenmundial.com/que-son-sanciones-internacionales/ [Accessed: 2022]

[2] "Reporting de Sancionados" [Online]. Available: https://www.sas.com/content/dam/SAS/es_es/doc/other1/sas-consulting-web.pdf [Accessed: 2022]

[3] "Overwrite existing files with Python's wget?" [Online]. Available: https://stackoverflow.com/questions/63226700/overwrite-existing-files-with-pythons-wget [Accessed: 2022]

[4] "¿Cómo descargar un archivo csv desde internet con python 3?" [Online]. Available: https://es.stackoverflow.com/questions/120084/c%C3%B3mo-descargar-un-archivo-csv-desde-internet-con-python-3 [Accessed: 2022]

[5] "Operaciones con expresiones regulares" [Online]. Available: https://docs.python.org/es/3/library/re.html [Accessed: 2022]

[6] "How to use get_close_matches() in Python" [Online]. Available: https://www.educative.io/answers/how-to-use-getclosematches-in-python [Accessed: 2022]

[7] Power Data, *Ebook. Procesos ETL: La Base de la Inteligencia de Negocio* (PDF). [Online]. Available: https://docplayer.es/222676367-Ebook-procesos-etl-la-base-de-la-inteligencia-de-negocio.html [Accessed: 2022]

[8] "¿Qué es el Data Cleansing?" [Online]. Available: https://www.crehana.com/blog/data-analitica/data-cleansing/ [Accessed: 2022]

[9] "¿Qué es integración de datos?" [Online]. Available: https://www.powerdata.es/integracion-de-datos [Accessed: 2022]

[10] "¿Qué es una expresión regular?" [Online]. Available: https://www.ionos.es/digitalguide/paginas-web/creacion-de-paginas-web/regex/ [Accessed: 2022]

[11] Aurélien Géron, *Aprende Machine Learning con Scikit-Learn, Keras y Tensorflow.* O'Reilly, Anaya.

[12] "¿Qué es el Deep learning?" [Online]. Available: https://blog.bismart.com/diferencia-machine-learning-deep-learning [Accessed: 2022]

[13] "Codificación binaria" [Online]. Available: https://elmundodelosdatos.com/tecnicas-para-codificar-variables-categoricas-binaria-hashing/ [Accessed: 2022]

[14] "One Hot Encoding" [Online]. Available: https://interactivechaos.com/es/manual/tutorial-de-machine-learning/one-hot-encoding#:~:text=Una%20alternativa%20al%20Label%20Encoding,dejar%20las%20dem%C3%A1s%20con%200. [Accessed: 2022]

[15] *scikit-learn API* [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.impute.SimpleImputer.html?highlight=simpleimputer#sklearn.impute.SimpleImputer [Accessed: 2022]

[16] "Qué es la programación orientada a objetos" [Online]. Available: https://desarrolloweb.com/articulos/499.php#:~:text=La%20programaci%C3%B3n%20Orientada%20a%20objetos%20se%20define%20como%20un%20paradigma,los%20objetivos%20de%20las%20aplicaciones. [Accessed: 2022]

[17] *scikit-learn API* [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.MiniBatchSparsePCA.html?highlight=minibatch+pca#sklearn.decomposition.MiniBatchSparsePCA [Accessed: 2022]

[18] "Entonces, ¿qué es un DataFrame?" [Online]. Available: https://datacarpentry.org/python-ecology-lesson-es/02-starting-with-data/ [Accessed: 2022]

[19] "Python para todos (2): ¿Qué son los Jupyter Notebooks??" [Online]. Available: https://empresas.blogthinkbig.com/python-para-todos-2-jupyternotebook/ [Accessed: 2022]