Logotipo

Descripción generada automáticamente

**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

El resumen tiene entre 150-250 palabras. Resumir consiste en ofrecer información exhaustiva sobre cómo, dónde, cuándo y por qué se aplica el proyecto. Se realiza al finalizar el trabajo.

El resumen

* Establece el tema tratado en el proyecto
* Resume los problemas planteados y las aportaciones más importantes del proyecto
* Resume las conclusiones más importantes del trabajo realizado

El resumen NO

* Da una información genérica
* Explica porque se realiza ese trabajo
* Se refiere a datos aportados en el texto del proyecto.

**Palabras clave:** hasta un máximo de 6 (DataFrame (mencionar la librería pandas y Python), Jupyter Notebook (mencionar la extension ipynb), scikit-learn),PowerBI

# ABSTRACT

Resumen en inglés.

**Key words:** Palabras clave en inglés

**AGRADECIMIENTOS**

En ocasiones se incluye este apartado para agradecer a aquellos que han ofrecido su ayuda en el desarrollo del trabajo, ya sea técnica o de otro tipo.

Índice

[RESUMEN 4](#_Toc113902626)

[ABSTRACT 4](#_Toc113902627)

[Capítulo 1. INTRODUCCIÓN 10](#_Toc113902628)

[1.1 Descripción del Problema 10](#_Toc113902629)

[1.1.1 Contexto Histórico 10](#_Toc113902630)

[1.1.2 Actualidad 10](#_Toc113902631)

[1.2 Motivación 10](#_Toc113902632)

[1.3 Conceptos Clave 11](#_Toc113902633)

[1.3.1 Técnicas de Procesamiento de Datos 11](#_Toc113902634)

[1.3.2 Conceptos Informáticos 14](#_Toc113902635)

[1.3.3 Técnicas de Análisis de Datos 16](#_Toc113902636)

[1.4 Estructura del proyecto (dejar para el final) 16](#_Toc113902637)

[Capítulo 2. Memoria Técnica 17](#_Toc113902638)

[2.1 Procesado de Datos 17](#_Toc113902639)

[2.1.1 Extracción y Selección de Datos 17](#_Toc113902640)

[2.1.2 Transformación y Limpieza de Datos 22](#_Toc113902641)

[2.1.3 Integración de Datos 27](#_Toc113902642)

[2.2 Modelos de Aprendizaje No Supervisado 29](#_Toc113902643)

[2.2.1 Preprocesamiento 29](#_Toc113902644)

[2.2.2 Modelos K-Means 32](#_Toc113902645)

[2.2.2.1 Modelo 1 33](#_Toc113902646)

[Capítulo 3. CONCLUSIONES Y FUTURAS LÍNEAS DE TRABAJO 35](#_Toc113902647)

[ANEXOS 36](#_Toc113902648)

[PRESUPUESTO Y TIEMPO 37](#_Toc113902649)

[BIBLIOGRAFÍA 38](#_Toc113902650)

Índice de Figuras

Este índice ha de existir tan sólo en el caso de que tengas más de 4 apoyos visuales (gráficos, figuras, ilustraciones, fotos, etc.) en tu memoria del proyecto.

Recuerda esta sección es optativa.

Índice de Tablas

Este índice ha de existir tan sólo en el caso de que tengas más de 4 tablas de datos en tu memoria del proyecto.

Recuerda esta sección es optativa.

# INTRODUCCIÓN

## 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.

### 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].

### 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].

## 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.

## Conceptos Clave

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

### Técnicas de Procesamiento de Datos

1. **Proceso ETL:** el proceso ETL es una parte de todo proceso de integración de datos. Su función tiene gran relevancia, ya que completa el resultado del desarrollo de aplicaciones y sistemas imprimiendo la cohesión necesaria [7].

La palabra ETL corresponde a las siglas en inglés de:

* Extraer: extract.
* Transformar: transform.
* Y Cargar: load.

Gracias a los procesos ETL es posible que cualquier organización:

* Mueva datos desde una o múltiples fuentes.
* Reformatee esos datos y los limpie, cuando sea necesario.
* Proceda a su carga en otro lugar, como puede ser una base de datos, un data mart o un data ware-house.
* Analice esos datos una vez alojados en destino.
* O los emplee en otro sistema operacional para apoyar un proceso de negocio, cuando ya están cargados en su ubicación definitiva.

Es necesario conocer el funcionamiento y claves de cada etapa del proceso ETL. A continuación, se resumen los aspectos más importantes de cada uno de estos procesos [7]:

**Proceso de extracción:** Extraer 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:** La fase de transformación de un proceso de ETL aplica una serie de reglas de negocio o funciones sobre los datos extraídos para convertirlos en datos que serán cargados. Estas directrices pueden ser declarativas, pueden basarse en excepciones o restricciones, pero, para potenciar su pragmatismo y eficacia, hay que asegurarse de que sean:

* Declarativas.
* Independientes.
* Claras.
* Inteligibles.
* Con una finalidad útil para el negocio.

**Proceso de carga:** En esta fase, los datos procedentes de la faseanterior (fase de transformación) son cargadosen el sistema de destino. Dependiendo de losrequerimientos de la organización, este procesopuede abarcar una amplia variedad deacciones diferentes. Por ejemplo, en algunasbases de datos será necesario sobrescribir lainformación antigua con nuevos datos mientrasque en otras, bastará con resumir lastransacciones y almacenar un promedio de lamagnitud considerada.

1. **Limpieza de datos:** El Data Cleansing o la limpieza de datos en una empresa es el proceso de corregir o eliminar datos formateados incorrectamente, duplicados o incompletos dentro de un gran conjunto de datos [8].

El Data Cleansing 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?

1. **Integración de datos:** La integración es el proceso que permite combinar datos heterogéneos de muchas fuentes diferentes en la forma y estructura de una única aplicación. Esto facilita que diferentes tipos de información, tales como matrices de datos, documentos y tablas, sean fusionados por usuarios, organizaciones y aplicaciones para un uso personal, de procesos de negocio o de funciones [9].

La integración soporta el procesamiento analítico de grandes conjuntos de datos alineando, combinando y presentando cada conjunto de informaciones de departamentos organizacionales y fuentes de datos remotas y externas, para cumplir con los objetivos del integrador [9].

### Conceptos Informáticos

1. **Expresiones regulares o regular expressions (regex):** Las regex son cadenas de caracteres basadas en reglas sintácticas que permiten describir secuencias de caracteres. Así, forman parte de los lenguajes regulares, los cuales son un subgrupo de los lenguajes formales, de gran importancia para la tecnología de la información y, especialmente, para el desarrollo de software [10].

Un claro ejemplo del uso de regex en la tecnología de la información es la función de buscar y reemplazar de los editores de texto, la cual fue implementada por primera vez en los años 60 por el pionero en las ciencias de la computación, Ken Thompson, uno de los desarrolladores del sistema operativo UNIX, en el editor de texto por línea QED y posteriormente en sus sucesores. Esta función permite buscar determinadas secuencias de caracteres en los textos y, si se desea, reemplazarlas por otra secuencia de caracteres cualquiera [10].

1. **Machine Learning & Deep Learning:** El Machine Learning 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. De hecho, se puede describir como la nueva evolución del Machine Learning. Se trata de un algoritmo automático que imita la percepción humana inspirada en nuestro cerebro y la conexión entre neuronas. El Deep Learning es la técnica que más se acerca a la forma en la que aprendemos los humanos [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.

1. **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.

1. **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**, similar a los métodos.

### Técnicas de Análisis de Datos

1. **Datamart en estrella:**

## Estructura del proyecto (dejar para el final)

Una buena organización de tu memoria ayuda a su lectura y refleja mejor la metodología de trabajo usada. Normalmente consta de objetivos, metodología, resultados y, conclusiones y futuras líneas de trabajo. Cuenta aquí más o menos en dos líneas por cada capítulo, el contenido de dicho capítulo.

NO OLVIDES:

* Numerar las páginas.
* Numerar las figuras
* Numerar las tablas
* Justificar los márgenes.
* Destacar los apartados dentro de cada capítulo
* Cuida el FORMATO: usa siempre el mismo tipo de letra, el mismo tamaño para el texto, el mismo tamaño para los apartados, el mismo tamaño para los títulos de los capítulos,…
* Si te dan una guía de estilo con tamaños de letra, interlineados, márgenes etc, **respétala**
* Destaca lo que creas oportuno utilizando las **negritas** y las *cursivas*.

# 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 esta sección se tratarán los problemas, cambios e imprevistos en la consecución de estos objetivos para finalmente presentar los resultados a los que se ha llegado.

## 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.

### 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 02en 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.

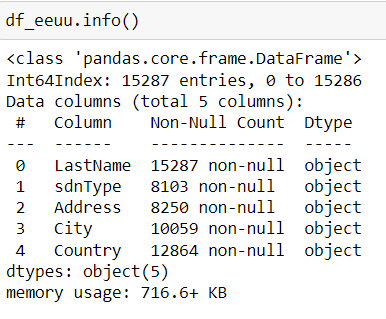


Figura 1: Información de metadatos de la tabla df\_eeuu.

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 que la mayoría de sancionados se corresponden a una empresa o asociación (fig. 2).

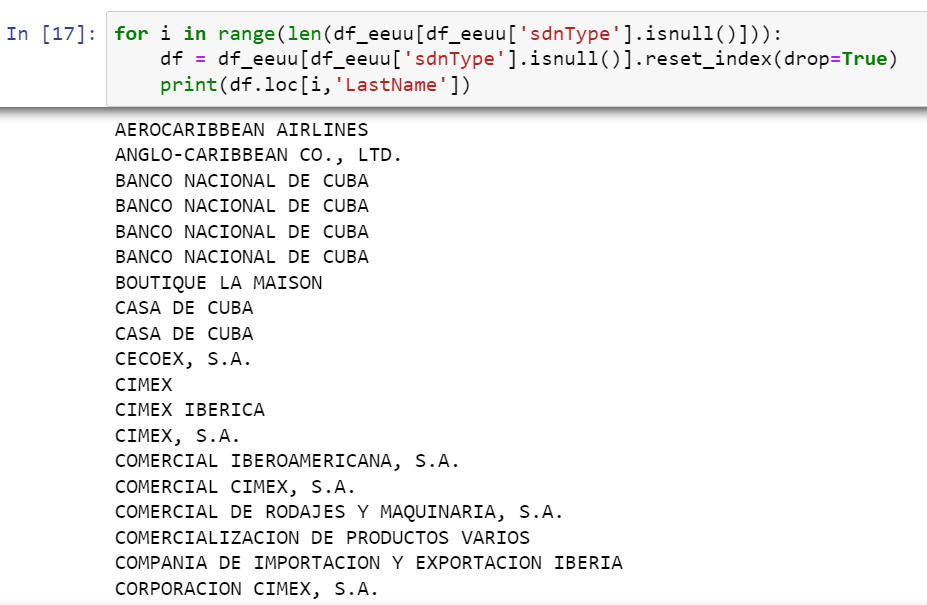


Figura 2: Bucle for para conocer los valores de la columna 'LastName' en los que 'sdnType' tiene valor nulo.

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.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

*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 (fig. 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.

Tabla

Descripción generada automáticamente

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 un gráfico de barras (fig. 5):

Gráfico

Descripción generada automáticamente

Figura 5: Gráfico de barras de la cardinalidad para cada valor de la columna 'Entity\_SubjectType\_ClassificationCode' de la tabla df\_ue

Como se aprecia en la figura 5, 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.

### 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.

Una vez aplicadas las transformaciones, 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.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

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.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

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'.

Finalmentese 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 (fig. 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.

Tabla

Descripción generada automáticamente

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* (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.

Tabla

Descripción generada automáticamente

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.

Tabla

Descripción generada automáticamente

Figura 10: Tabla transformada final df\_ue\_t3

Finalmentese 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.

### 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 (fig. 11).

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza mediaDe este modo, se tiene toda la información centralizada en un único dataset depurado que puede cargarse al script de preprocesamiento y aplicación de modelos ML, en PowerBI o en cualquier otra herramienta de análisis de datos que se quiera emplear.

Figura 11: Tabla final de sancionados

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.
* 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 será leído por el programa de aplicación y evaluación del modelo ML y cargado en PowerBI para la generación del dashboard.

## Modelos de Aprendizaje No Supervisado

Como algoritmos no supervisados se utilizarán K-Medias y DBSCAN para agrupar los registros y PCA para reducir la dimensionalidad de las variables. Se aplicará también alguna clase capaz de eliminar outliers. Como métrica para evaluar la calidad del modelo se calculará el coeficiente de silueta. Con el objetivo de entender mejor los agrupamientos, la evaluación de los modelos se apoyará de gráficas y diagramas.

Para el preprocesamiento de los datos y aplicación de los distintos modelos de aprendizaje no supervisado, se comienza por cargar los datos en 'datos\_entrada\_script\_04\_y\_PowerBI' dentro del script 04\_modelos\_ML\_no\_supervisado.ipynb y se lee como DataFrame.

Se definirán varios modelos de aprendizaje no supervisado y se comparará la eficacia de cada uno de ellos.

### Preprocesamiento

En este apartado se describen los pasos previos a la aplicación de los algoritmos de aprendizaje no supervisado para agrupamiento.

Selección de Variables

Al aplicar la función info() sobre el DataFrame se ve que hay demasiados valores nulos para las columnas ‘Address’, ‘Country’ y ‘City’. Por estudios llevados a cabo en apartados anteriores, se sabe que la mayoría de estos nulos proceden de la tabla de la Unión Europea.

Se llega a la conclusión de que el dataset de la Unión Europea es demasiado malo para aplicarle un algoritmo de Machine Learning. Por ello, finalmente se opta por utilizar únicamente el dataset de Estados Unidos para el entrenamiento del algoritmo de aprendizaje no supervisado.

Para seleccionar únicamente los registros de Estados Unidos, se filtra por aquellas filas para las que la columna Origin tenga valor ‘EEUU’. A partir del resultado de aplicar la función describe() sobre la tabla filtrada, se puede conocer la cardinalidad de cada variable:

* La variable Name presenta un 68.55 % de valores únicos sobre el total de valores no nulos.
* La variable EntityType presenta un 0.03 % de valores únicos sobre el total de valores no nulos.
* La variable Address presenta un 86.26 % de valores únicos sobre el total de valores no nulos.
* La variable Country presenta un 1.39 % de valores únicos sobre el total de valores no nulos.
* La variable Origin presenta un 0.01 % de valores únicos sobre el total de valores no nulos.
* La variable City presenta un 20.44 % de valores únicos sobre el total de valores no nulos.

A mayor porcentaje, mayor es la cardinalidad. No interesan las variables con un porcentaje de valores únicos alto porque esto podría acarrear problema de rendimiento y de calidad del modelo. Por ello, inicialmente se eliminaron las variables 'Name' y 'Address' del dataset para entrenar el modelo.

Sin embargo, el hecho de eliminar estas dos variables hacía que el dataset tuviera muchos valores duplicados. De modo que se perdía gran cantidad de información para el agrupamiento y el modelo de clustering solo agrupaba por los valores únicos.

Para evitar esto, se volvió a introducir al dataset de entrenamiento la variable 'Name' y se eliminaron los duplicados. La variable 'Address' se eliminó definitivamente porque además de presentar alta cardinalidad también presenta muchos valores nulos.

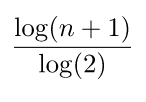
La variable Origin se puede eliminar porque ahora solo tiene un valor, por lo que no aporta información.

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. Los pasos que sigue este método para codificar son los siguientes [13]:
* Primeramente, cada categoría única de nuestra variable categórica se convierte en un número ordinal sin tener en cuenta ningún orden. Así, este número irá desde el 1 hasta el número total de categorías únicas que tengamos.
* Después, estos valores ordinales se transforman a su código binario. De esta manera, la categoría que le corresponda un valor de 10 se convertiría al código binario 1010.
* Finalmente, cada dígito binario se separa en distintas columnas, una por cada dígito, por lo que para el código binario 1010 necesitaríamos 4 columnas.

Para calcular el número total de columnas que se crearán tras este procedimiento, podemos aplicar la siguiente formula, cuyo resultado deberemos redondear hacia arriba [13]:



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 son los siguientes:

* **SimpleImuter:** 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 [11].

Antes de poder proyectar el conjunto de entrenamiento en un hiperplano con menos dimensiones, primero se elige el hiperplano más adecuado [11].

Parece razonable seleccionar el hiperplano que preserva la máxima cantidad de varianza, ya que lo más probable es que pierda menos información que las otras proyecciones. 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].

La librería de scikit-learn tiene varias clases para aplicar PCA. 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].
* **MiniBatchSparsePCA:** También presente en sklearn.decomposition. El parámetro ajustable *n\_clusters* permite seleccionar el número de dimensiones con las que se queda el algoritmo. El parámetro *alpha* permite controlar la dispersión de los componentes seleccionados [17].

### Modelos K-Means

El algoritmo K-Means de aprendizaje no supervisado para agrupamiento es el primero en ser utilizado. Se desarrollan tres modelos distintos utilizando K-Means. Estos se diferencian en el preprocesamiento de los datos previo a su entrada en el algoritmo.

Se utiliza la clase **KMeans** del paquete sklearn.cluster para agrupar los registros. Su funcionamiento es el siguiente [11]:

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 puede variar entre -1 y +1. Un coeficiente cercano a +1 significa que la instancia está muy metida en su propio grupo y lejos de otros grupos, mientras que un coeficiente cercano a 0 significa que está cerca del límite de un grupo y, por último, un coeficiente cercano a -1 indica que la instancia puede haberse asignado al grupo equivocado [11].

Se utiliza la función silhouette\_score() para conocer el coeficiente medio de silueta para cada número de grupos seleccionado.

#### Modelo 1

**Preprocesamiento**

La lógica del preprocesamiento en este modelo aparece en la figura 12.

Se trata de un **Pipeline** de 3 pasos. Pipeline es una clase de scikit-learn que permite encadenar varias transformaciones sobre un set de datos, poniendo como entrada del siguiente paso la transformación realizada por el anterior.

Primer paso (encoder)

Se aplica **ColumnTransformer**. Esta clase permite seleccionar la columna sobre la que se aplica la transformación de los datos. El primer transformador hace un Pipeline sobre la columna Name para encadenar dos transformaciones:

La primera de ellas aplica un codificador ordinal sobre ‘Name’ y la segunda (MinMaxScaler) establece un rango a los valores codificados.

El problema de aplicar OrdinalEncoder es que estamos tratando a la variable Name, que es categórica nominal, como ordinal al aplicarle un orden alfabético. Esto lleva a que a nombres con una posición alfabética más alta se les asignen valores más altos.

Para evitar que haya valores con números muy altos que sean más relevantes que los bajos, se aplica posteriormente MinMaxScaler. Esta clase escala cada valor a un rango de nuestra elección. Poniendo un rango entre valores pequeños, se pretende reducir el impacto de OrdinalEncoder sobre el modelo de agrupamiento.

El otro transformador dentro de ColumnTransformer es un codificador binario, el cual se encarga de codificar las variables EntityType, Country y City.

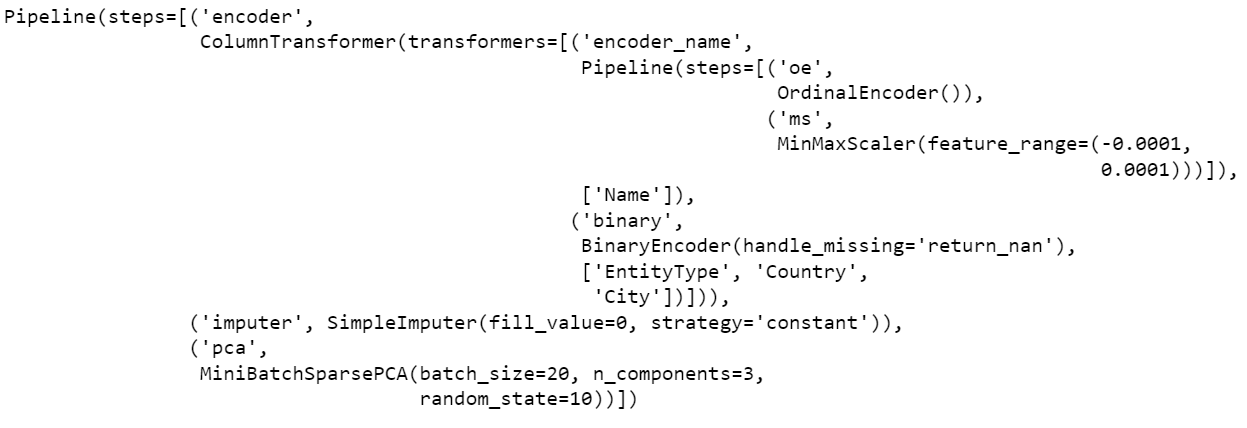


Figura 12: Lógica del código Python de preprocesamiento del modelo 1.

Segundo paso (imputer)

Se aplica un imputador simple que sustituye todos los valores nulos por un cero.

Tercer paso (pca)

Se aplica la clase MiniBatchSparsePCA para reducir la dimensionalidad a 3 variables.

**Agrupamiento**

El código de agrupamiento por el algoritmo K-Means para grupos de 2 a 7 aparece en la figura 13.

Se aplica un bucle for para pasarle al parámetro *n\_clusters* los valores enteros de 2 a 7. También se calcula el valor de la silueta media para cada selección de grupos.

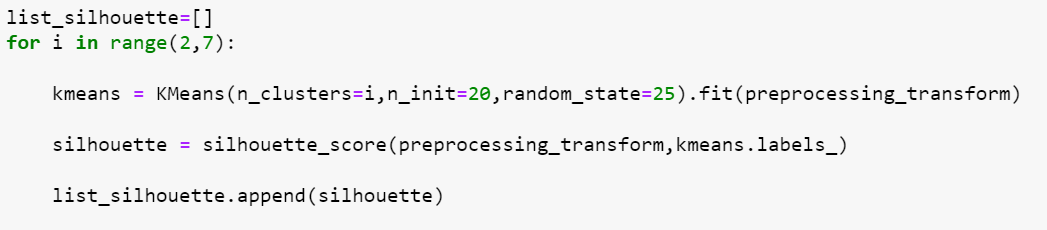


Figura 12: Código Python de aplicación del algoritmo K-Means sobre los datos tras su preprocesamiento.

Los valores del coeficiente de silueta se llevan a una lista para su posterior representación gráfica en la sección de Evaluación del Modelo.

**Evaluación del modelo**

Una vez aplicado el algoritmo K-Means, se crean representaciones gráficas a partir de los datos recopilados para entender mejor el modelo y poder evaluar su calidad.

## Generación de Dashboards

* Comentar que solo con el nombre las ubicaciones podrían estar representádose mal en el mapa. Ya que existen distintas regiones geográficas con el mismo nombre.
* Interfaz de usuario gráfica, Texto, Aplicación

  Descripción generada automáticamente
* Explicar el desbalance entre los sancionados representados en el piechart y en el mapa geográfico
* Dashboard:
* - Ranking países sancionados
* - Analizar diferencias entre proporciones EEUU y UE
* - Ver si hay sancionados presentes en las dos listas.

Modificaciones hechas al analizar datos con powerbi:

* Se descubre gracias a PowerBI que existían valores nulos para la columna 'Country' etiquetados como 'UNKNOWN'. Se debe modificar para cambiarlo a NaN como el resto de valores nulos.
* Revisando los datos con PowerBI se descubre que la expresión regular aplicada no ha conseguido eliminar algunos de los códigos que aparecen en la variables 'City'. Estos códigos son valores de códigos postales que no van acompañados del nombre de ninguna ciudad. Estos códigos deben ser sustituidos por NaN.

# CONCLUSIONES Y FUTURAS LÍNEAS DE TRABAJO

Es más que conveniente terminar un trabajo con unas conclusiones que sirvan de colofón al mismo. Caben en ellas tus impresiones, o destacar la importancia que tiene el tema, lo que has aprendido, o la trascendencia, etc.

Además las conclusiones tienen que referirse siempre a los objetivos generales y específicos empleados.

Es importante destacar en este apartado las líneas con las que se podría continuar el mismo.

* -COMENTAR EN CONCLUSIONES QUE LA TRADUCCION Y EL RESOLVEDOR DE MATCHES SON MUY LENTOS Y SE PLANTEÓ UTILIZAR SPARK PERO QUE HABRÍA QUE HABER CARGADO LOS SCRIPTS A RDD’S Y ADAPTAR LA LÓGICA DEL CÓDIGO AL DE LA APLICACIÓN DE BIG DATA.
* -COMENTAR EN FUTURAS LÍNEAS DE TRABAJO COMO EL HABER TENIDO VARIABLES NUMÉRICAS HABRÍA PODIDO MEJORAS LAPOSIBILIDADES DE LOS MODELOS ML. LAS VARIABLES NUMÉRICAS PODRÍAN GENERARSE, POR EJEMPLO, A PARTIR DE LA LATITUD Y LONGITUD DE LA UBICACIÓN.

# ANEXOS

Sirven para incluir documentación complementaria (planos, circuitos, códigos, especificaciones, hojas características, fichas explicativas, etc.)

Eso sí, cada comentario en una página independiente. Y ES QUE LOS ANEXOS SE ORGANIZAN ASÍ: CADA DOCUMENTO EN UNA PÁGINA INDEPENDIENTE.

# 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 (PowerBI) 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 a 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  Internet | 4.99 – 15 €/mes  31 – 33 €/mes | Sí  No |  |
|  |  |  |  |  |

FALTA DIAGRAMA DE GANTT

# 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]