

PROYECTO FINAL MÁSTER
CLASIFICADOR DOCUMENTOS MÉDICOS HOPE
2020 - 2021

Ruben Vasallo Gonzalez

1 de septiembre de 2020

Índice general

1.	2
1.1. Resumen	2
1.2. Abstract	2
1.3. Keywords	2
2. Introducción	3
2.1. Definición del proyecto	3
2.2. Estado del arte	3
3. Objetivos del Máster	4
3.1. Objetivo principal	4
3.2. Objetivos secundarios	4
4. Metodología	5
4.1. Reuniones con el cliente	5
4.2. Extracción de los datos	5
4.2.1. Lectura de los datos	8
4.2.2. Conversión a formato columnar	9
4.3. Procesado de los datos	12
4.3.1. Análisis de los datos	12
4.3.2. Análisis de componentes principales	14
4.4. Enriquecimiento de los datos. Aproximación por Vecinos más próximos (K-NN)	19
4.5. Modelos Predictivos	20
4.5.1. Regresión logística ' <i>Logistic regression</i> '	20
4.5.2. Bosques Aleatorios ' <i>Random Forest</i> '	20
4.5.3. Maquinas de Vector Soporte ' <i>Support Vector Machines</i> '	20
4.6. Resultados	21
5. Conclusiones	22
6. Bibliográfica	23
Índice de figuras	24
7. Anexos	25

Capítulo 1

1.1. Resumen

El proyecto nace de la necesidad de poder disponer de una manera sencilla e inmediata, artículos médicos catalogados según los síntomas de pacientes, pudiendo hacer un *ranking* de más o menos interés en función del *feedback* aportado por los profesionales sanitarios sobre artículos relacionados con esos *síntomas*.

1.2. Astract

TODO

1.3. Keywords

clasificador articulos medicos, PCA, KNN, Regresion Logistica, Random Forest, SVM

Capítulo 2

Introducción

2.1. Definición del proyecto

El proyecto que aquí se presenta nace de la necesidad por parte del *proyecto HOPE* de clasificar y recomendar resultados sobre estudios clínicos de confianza y que estén actualizados. En Internet existe muchísima información sobre medicina y salud y no siempre toda es de fiar.

El proyecto HOPE (que significa *Health Operations for Personalized Evidence* en inglés) nace de la necesidad de ayudar a los profesionales sanitarios a encontrar la información que necesitan de la manera más rápida y fácil posible. Existe infinidad de información médica en Internet de miles de proyectos de investigación médica y esto hace que, muchas veces sea complicado encontrar la información sobre ensayos médicos para tratar información. En el ámbito de la medicina el tiempo perdido puede costar vidas y es un precio demasiado elevado a pagar, tanto a nivel económico como emocional.

Actualmente existen bases de datos de confianza en donde los profesionales sanitarios y el público en general puede buscar informes y ensayos sobre estudios clínicos desarrollados anteriormente, pero no siempre es fácil o rápido encontrar estos resultados.

El proyecto HOPE es un sistema basado en inteligencia artificial para identificar los datos claves de casos clínicos registrados en la Historia Clínica Electrónica, en base a los cuales realiza una búsqueda única por paciente para proporcionar al profesional sanitario recomendaciones de tratamientos, estudios de investigación, información para el paciente, todo en base a registros de fuentes científicas de información. En este proyecto, profesionales sanitarios de todo el mundo puede consultar en una base de datos informes médicos relacionados con los síntomas que puedan tener sus pacientes y ver que otros tratamientos han dado resultado. Todo y con eso, el sistema no siempre devuelve los artículos más relevantes o actualizados por lo que, no siempre la información consultada es útil.

En este ámbito, los profesionales sanitarios pueden valorar si la información recibida ha sido útil o no respecto a la búsqueda que han realizado, por lo que con ese *feedback*, se pretende mejorar el sistema actual complementándolo con un modelo clasificador capaz de ayudar al actual a entregar realmente los artículos útiles basándose en el *feedback* que los profesionales sanitarios dan al sistema.

2.2. Estado del arte

Recomendadores que existen actualmente:

Capítulo 3

Objetivos del Máster

3.1. Objetivo principal

OP - Poder recomendar al profesional sanitario cuales son los artículos más útiles que pueden ayudar en el tratamiento del paciente, en base a los síntomas que este tiene, pudiendo realizar un *ranking* de mas interés a menos.

3.2. Objetivos secundarios

Para poder cumplir con el objetivo principal [OP1](#), desglosaremos los siguientes objetivos secundarios:

OS1 - Extraer la información de la base de datos y tratarla para quedarnos solo con la que consideramos valida.

OS2 - Hacer un análisis de componentes principales (estudio de que atributos son relevantes para alcanzar el objetivo).

OS3 - Enriquecer de los datos (*data augmentation*) prediciendo los resultados que no están indicados si son relevantes o no. Aproximación por Vecinos más próximos (*K-Nearest-Neighbor*).

OS4 - Predecir los resultados usando el algoritmo de aprendizaje supervisado para clasificación llamado Regresión logística '*Logistic regression*'.

OS5 - Predecir los resultados usando el algoritmo de aprendizaje supervisado para clasificación llamado Bosques Aleatorios '*Random Forests*'.

OS6 - Predecir los resultados usando el algoritmo de aprendizaje supervisado para clasificación llamado Máquinas de vector soporte '*Support Vector Machines*'.

Capítulo 4

Metodología

4.1. Reuniones con el cliente

Para poder comprender y abordar con éxito el **objetivo principal** se realizaron 4 reuniones en donde el cliente expuso el **problema** a abordar y el origen de los datos para poder realizar el estudio.

En estas reuniones se pudo observar que los datos facilitados por el usuario requerían de una limpieza y tratamiento para poder cumplir el objetivo principal, ya que muchas observaciones tenían información poco relevante que podía generar ruido.

Realizando un primer análisis visual, se detectó que los datos aportados por el cliente eran insuficientes para completar el **OP1**, ya que solo se disponía de la información respecto de si un artículo había sido útil o no, pero no se disponía de la información suficientemente detallada para saber si había sido muy útil o poco útil para poder llegar a realizar un *ranking*. El cliente nos comenta que en el momento actual no dispone de ese nivel de detalle y se acuerda con el que, se realizara una aproximación para indicar si un artículo es útil o no dejando para mas adelante la opción de poder realizar *rankings* si se consigue ese nivel de detalle por parte del cliente.

También se pudo comprobar que el cliente disponía de un volumen de observaciones bajo por lo que se planteó la posibilidad de, o intentar obtener más observaciones facilitadas por el cliente, o intentar enriquecer las observaciones actuales generando nuevos datos por aproximación a los reales.

Finalmente se decidió estudiar si era viable generar nuevos valores por aproximación, debido a que en el momento en que se trató el problema, el cliente no podía facilitar más datos. Si a lo largo del estudio, el cliente conseguía facilitar nuevas observaciones, estas serían añadidas al estudio para aproximar mejor la solución final.

4.2. Extracción de los datos

Para cumplir con el **OP1** mostramos los pasos que hemos seguido para extraer y procesar los datos:

El Origen de los datos se encuentra en una Base de datos SQL distribuida en dos tablas, que pasamos a detallar a continuación:

En la primera tabla llamada *fed_hope_sugerencia*, encontraremos la sugerencia que dio el programa HOPE en base a los parámetros que introdujo el profesional sanitario, almacenado en el atributo pedido y la respuesta que dio el programa, almacenado en el atributo respuesta. Todos los datos son almacenados en formato documento json.

En la figura 4.1 mostramos los atributos de la tabla *fed_hope_sugerencia*.

Table: fed_hope_sugerencia

Select data Show structure Alter table New item

Column	Type	Comment
id	int(11)	
pedido	longtext NULL	
respuesta	longtext NULL	

Figura 4.1: Visualización de los atributos de la tabla *fed_hope_sugerencia*

Si analizamos el **atributo pedido**, podemos observar, tal y como se muestra en la figura 4.2, varios atributos haciendo referencia a los síntomas que consulta el profesional sanitario.

```
{
  "data": {
    "type": "emr--em",
    "attributes": {
      "name": null,
      "affected_organ": "",
      "age": "75",
      "diagnostic_main": "FISTULA PERITONEAL",
      "gender": "male",
      "medical_history": "Paciente de 75 años con antecedentes de gastrectomía total por adenocarcinoma gástrico que intercurrió con eventración y posterior formación de fístula entero-atmosférica. Actualmente cursa postoperatorio de resección intestinal mu00e1 eventroplástica con colocación de malla. Al examen impresionado en regular estado general, lúcido, hemodinámicamente estable, sin signos de falla de bomba. Regular entrada de aires, rales crepitantes bilaterales. Abdomen blando, depresible, levemente doloroso a la palpación profunda. Herida cubierta por apósitos estériles. Catarsis positiva. Diuresis positiva.\n\nPROBLEMAS ACTIVOS:\n- POP resección intestinal mu00e1 eventroplástica: Paciente clínicamente estable, hemodinámicamente compensado. Refiere buena tolerancia al dolor. Afebril hace 72 hs. Cumple 4to día de tratamiento con piperacilina tazobactam por neumonía broncoaspirativa con aislamiento de E.coli BLEE. Hemocultivos vienen negativos. TAC de tórax y abdomen informa: Consolidación bibasal bilateral mu00e1 derrame pleural bilateral. Colección laminar posterior a ambos mu00fasculos rectos de 10x2x0.4 cm. y otra colección en herida quirúrgica de pared de 10x2.8x1.2 cm. Se da aviso a cirujano tratante. Persiste con estado nauseoso, por lo que continúa con antieméticos reglados. \nEn aislamiento de contacto por germen multirresistente. \nSe da informe. Control evolutivo."
    }
  }
}
```

Figura 4.2: Muestra de una observación del atributo pedido

Si analizamos el **atributo respuesta**, podemos observar entre otros datos, el listado de artículos médicos sugeridos relacionados con los síntomas descritos por el profesional sanitario. Esta respuesta es muy amplia pero entre todos los atributos, podemos observar un listado de identificadores de artículos, con sus fechas de revisión de estos, y unas palabras claves descriptivas para esos artículos.

A continuación mostramos en la figura 4.3 una pequeña parte del contenido de una observación del atributo respuesta.

```
{
  "data": {
    "type": "emr--emr",
    "id": "ef6d63fb-afe8-4650-8ab8-d4d75edd4fe5",
    "attributes": {
      "id": 376,
      "uuid": "ef6d63fb-afe8-4650-8ab8-d4d75edd4fe5",
      "language": "es",
      "name": null,
      "status": true,
      "created": 1559052244,
      "changed": 1559052244,
      "affected_organ": null,
      "age": 75,
      "clinicaltrials": {
        "query": [],
        "trials": [],
        "diagnostic_main": "FISTULA PERITONEAL",
        "diagnostic_main_mesh_terms": null,
        "gender": "male",
        "history_date": null,
        "medical_history": "Paciente de 75 años con antecedentes de gastrectomía total por adenocarcinoma gástrico que intercurrió con eventración y posterior formación de fístula entero-atmosférica. Actualmente cursa postoperatorio de resección intestinal muéls eventroplasto con colocación de malla. Al examen impresionado en regular estado general, muéls, hemodinámica estable, sin signos de falla de bomba. Regular entrada de aire, rales crepitantes bilaterales. Abdomen blando, depresible, levemente doloroso a la palpación profunda. Herida cubierta por apósitos estériles. Catarsis positiva. Diuresis positiva."
      }
    }
  },
  "problems": {
    "type": "emr--emr",
    "id": "ef6d63fb-afe8-4650-8ab8-d4d75edd4fe5",
    "attributes": {
      "id": 376,
      "uuid": "ef6d63fb-afe8-4650-8ab8-d4d75edd4fe5",
      "language": "es",
      "name": null,
      "status": true,
      "created": 1559052244,
      "changed": 1559052244,
      "affected_organ": null,
      "age": 75,
      "clinicaltrials": {
        "query": [],
        "trials": [],
        "diagnostic_main": "FISTULA PERITONEAL",
        "diagnostic_main_mesh_terms": null,
        "gender": "male",
        "history_date": null,
        "medical_history": "Paciente de 75 años con antecedentes de gastrectomía total por adenocarcinoma gástrico que intercurrió con eventración y posterior formación de fístula entero-atmosférica. Actualmente cursa postoperatorio de resección intestinal muéls eventroplasto con colocación de malla. Al examen impresionado en regular estado general, muéls, hemodinámica estable, sin signos de falla de bomba. Regular entrada de aire, rales crepitantes bilaterales. Abdomen blando, depresible, levemente doloroso a la palpación profunda. Herida cubierta por apósitos estériles. Catarsis positiva. Diuresis positiva."
      }
    }
  },
  "problems_active": {
    "type": "emr--emr",
    "id": "ef6d63fb-afe8-4650-8ab8-d4d75edd4fe5",
    "attributes": {
      "id": 376,
      "uuid": "ef6d63fb-afe8-4650-8ab8-d4d75edd4fe5",
      "language": "es",
      "name": null,
      "status": true,
      "created": 1559052244,
      "changed": 1559052244,
      "affected_organ": null,
      "age": 75,
      "clinicaltrials": {
        "query": [],
        "trials": [],
        "diagnostic_main": "FISTULA PERITONEAL",
        "diagnostic_main_mesh_terms": null,
        "gender": "male",
        "history_date": null,
        "medical_history": "Paciente de 75 años con antecedentes de gastrectomía total por adenocarcinoma gástrico que intercurrió con eventración y posterior formación de fístula entero-atmosférica. Actualmente cursa postoperatorio de resección intestinal muéls eventroplasto con colocación de malla. Al examen impresionado en regular estado general, muéls, hemodinámica estable, sin signos de falla de bomba. Regular entrada de aire, rales crepitantes bilaterales. Abdomen blando, depresible, levemente doloroso a la palpación profunda. Herida cubierta por apósitos estériles. Catarsis positiva. Diuresis positiva."
      }
    }
  }
}
```

Figura 4.3: Ejemplo de contenido del atributo respuesta de una observación

En la segunda tabla llamada *fed_hope_sugerencia_feedback*, encontraremos, tal y como se muestra en la figura 4.4, la opinión *feedback* (que utilidad ha tenido la información por parte del profesional sanitario) de la información recibida dado un artículo en concreto en una búsqueda en concreto. Esta información se relaciona con la tabla *fed_hope_sugerencia* a través del atributo *fed_hope_sugerencia_id*.

En esta tabla, esta representada la opinión *feedback* del profesional sanitario en el atributo *utilidad*, que denota un valor 0 para los artículos que han sido poco útiles respecto a la búsqueda realizada y 1 para los artículos que si han sido útiles.

Table: fed_hope_sugerencia_feedback

Select data Show structure Alter table New item

Column	Type	Comment
id	int(11)	
articulo	varchar(255) NULL	
utilidad	int(11) NULL	
comentario	varchar(255) NULL	
fed_hope_sugerencia_id	int(11) NULL	

Figura 4.4: Visualización de los atributos de la tabla *fed_hope_sugerencia_feedback*

4.2.1. Lectura de los datos

Para extraer los datos de la base de datos nos ayudaremos de las librerías *sqlalchemy* y *pymysql* programadas en lenguaje *python* que nos permitirá acceder a la información almacenada en una base de datos *MySQL* y devolvérsela en formato *dataframe*, un formato que nos permite entre otras cosas, realizar transformaciones de los datos para conseguir nuestro objetivo final.

Este formato es interpretable por la librería *pandas* y *numpy*, dos librerías programadas en lenguaje *python*, muy comunes en el ámbito de la ciencia del dato, que nos facilitara entre otras cosas, poder hacer operaciones matemáticas con los datos de manera eficiente. A continuación mostramos en la figura 4.5 el código utilizado para extraer los datos de la tabla *fed_hope_sugerencia*.

Import data from DB.

```
In [1]: # pip install pymysql
        from sqlalchemy import create_engine
        import pymysql
        import pandas as pd
        import numpy as np

In [2]: dbConnectionURL = 'mysql+pymysql://root:hope@mysql-master/hope'
        dbConnection = create_engine(dbConnectionURL)

        df = pd.read_sql('SELECT id, pedido, respuesta FROM fed_hope_sugerencia', con=dbConnection)

In [3]: df.head(10)

Out[3]:
```

	id	pedido	respuesta
0	29	{ "data": { "type": "emr-em", "attributes": { "name": ...	{ "data": { "type": "emr-em", "id": "ef6d63fb-afe8...
1	30	{ "data": { "type": "emr-em", "attributes": { "name": ...	{ "data": { "type": "emr-em", "id": "0b8a1cc8-ce17...
2	31	{ "data": { "type": "emr-em", "attributes": { "name": ...	{ "data": { "type": "emr-em", "id": "25733e18-3245...
3	32	{ "data": { "type": "emr-em", "attributes": { "name": ...	{ "data": { "type": "emr-em", "id": "40320232-7510...
4	33	{ "data": { "type": "emr-em", "attributes": { "name": ...	{ "data": { "type": "emr-em", "id": "7686d89e-1c8e...
5	34	{ "data": { "type": "emr-em", "attributes": { "name": ...	{ "data": { "type": "emr-em", "id": "d94d7c78-9941...
6	35	{ "data": { "type": "emr-em", "attributes": { "name": ...	{ "data": { "type": "emr-em", "id": "0a14cc6b-af7b...
7	36	{ "data": { "type": "emr-em", "attributes": { "name": ...	{ "data": { "type": "emr-em", "id": "245bb87d-b52c...
8	37	{ "data": { "type": "emr-em", "attributes": { "name": ...	{ "data": { "type": "emr-em", "id": "1ad05206-0416...
9	38	{ "data": { "type": "emr-em", "attributes": { "name": ...	{ "data": { "type": "emr-em", "id": "a0f8dabe-a795...

Figura 4.5: Lectura de los datos

Aplicaremos los mismos pasos para leer la información del feedback de los profesionales sanitarios de la tabla *fed_hope_sugerencia_feedback*

4.2.2. Conversión a formato columnar

Para poder trabajar con los datos, necesitaremos que estos estén en formato columnar (tabla relacional) por lo que necesitaremos convertir los datos de estos *json* en tablas relacionales (A esta acción se le conoce como *flattening* o aplanar).

Este paso consiste en coger cada uno de los atributos que tiene el *json* y convertirlos en columnas de una tabla, añadiendo los valores. Si el *json* tiene varios niveles, este proceso añadirá tantas columnas como niveles tenga el *json*, siempre que todas las observaciones del *json* tengan el mismo formato. Este caso se nos cumple para las observaciones del atributo Pedido. No es así para las observaciones del atributo respuesta en el que tendremos que hacer un tratamiento especial que detallaremos posteriormente.

Cuando se analizan los datos recuperados, se detecta que estos, contienen caracteres que informan de los saltos de línea o tabulación. Estos caracteres pueden ser mal interpretados a la hora de leer los datos de los documentos en formato *json* por lo que será necesario eliminarlos.

• *Flattening* del atributo pedido

Para realizar la acción de *flattening* en el atributo pedido, nos ayudaremos de la funcionalidad *json_normalize* del paquete *pandas* que realiza esta acción. A continuación mostramos en la figura 4.6 el código utilizado para el atributo pedido.

Flattening JSON

```
In [5]: import ast
import json
from pandas import read_json, json_normalize #package for flattening json in pandas df
#https://stackoverflow.com/questions/39899005/how-to-flatten-a-pandas-dataframe-with-some-columns-as-json

pd.options.display.max_columns = None
#pd.options.display.max_rows = None
```

```
In [6]: # Flatten column "Pedido"

pedidosData = json_normalize(df['pedido'].apply(json.loads).tolist()).add_prefix('pedido.')
```

Out[6]:

	pedido.data.type	pedido.data.attributes.name	pedido.data.attributes.affected_organ	pedido.data.attributes.age	pedido.data.attributes.diagnostic_main	pedido
0	emr--em	None		75	FISTULA PERITONEAL	
1	emr--em	None		31	REHABILITACION NEUROLOGICA	
2	emr--em	None		76	INSUFICIENCIA CARDIACA	
3	emr--em	None		75	FISTULA PERITONEAL	
4	emr--em	None		31	REHABILITACION NEUROLOGICA	
...
119	emr--em	None		74	DIFICULTAD RESPIRATORIA	
120	emr--em	None		48	REHABILITACION NEUROLOGICA	
121	emr--em	None		40	REHABILITACION NEUROLOGICA	
122	emr--em	None		43	TEP	
123	emr--em	None		37	DOLOR ABDOMINAL	

124 rows x 7 columns

Figura 4.6: *Flattening* del atributo pedido.

• *Flattening* del atributo respuesta

Debido a que la información almacenada en el documento *json*, en el atributo respuesta es muy compleja (debido a que esta contiene diferentes documentos con diferentes niveles de información) como se puede apreciar en el apartado X, no podemos aplanar la información directamente como hemos hecho con el atributo pedido. Por lo que tenemos que analizar que información nos interesa recoger para enriquecer el conjunto de datos.

Después de analizar el documento, y ayudarnos del conocimiento del cliente, vemos que los atributos más interesantes son los que hacen referencia al identificador del artículo, las palabras claves asociadas al artículo por parte de la api pubmed y el mes y año de la revisión del artículo. Para recoger esta información nos crearemos una función que acceda directamente a estos atributos dado una observación. Después ejecutaremos esa función para cada observación ayudándonos de la función *apply*. A continuación mostramos este proceso en la figura 4.7.

```
In [7]: # Flattenin column "respuesta"

def get_articles_from_respuesta(ld):
    jsonData = json.loads(ld)
    pubmedKeys = jsonData['data']['attributes']['pubmed_mt_opt']
    if pubmedKeys is None: pubmedKeys = []

    articles = list(jsonData['data']['attributes']['pubmed']['articles'])
    articlesIDs = []
    articlesRevisedYear = []
    articlesRevisedMonth = []
    for article in articles:
        articlesIDs.append(article['id'])
        articlesRevisedYear.append(article['revisedDate']['Year'])
        articlesRevisedMonth.append(article['revisedDate']['Month'])

    return dict({
        'articles': articlesIDs,
        'articlesRevisedYear': articlesRevisedYear,
        'articlesRevisedMonth': articlesRevisedMonth,
        'pubmed_keys': ','.join(pubmedKeys)
    })

respuestaData = json_normalize(df['respuesta'].apply(get_articles_from_respuesta).tolist()).add_prefix('respu'
a.')
respuestaData
```

Out[7]:

	respuesta.articles	respuesta.articlesRevisedYear	respuesta.articlesRevisedMonth	respuesta.pubmed_keys
0	[27395425, 28560554, 28641726, 26245344, 28942...	[2018, 2018, 2017, 2016, 2018, 2014, 2018, 201...	[01, 04, 12, 12, 06, 06, 09, 04, 01, 04]	Intestines,Therapeutics,Catharsis,Wounds and I...
1	[30210096, 27617939, 27210858, 26412482, 25487...	[2019, 2017, 2017, 2017, 2016, 2016, 2019, 201...	[03, 04, 08, 06, 06, 09, 02, 03, 01, 11]	Back,Wounds and Injuries,Catheterization,Rest,...
2	[21067951, 27616270, 27532500, 28426556, 27495...	[2011, 2017, 2017, 2019, 2017, 2009, 2017, 201...	[03, 07, 05, 01, 07, 03, 06, 05, 03, 03]	Heart Murmurs,Intestines,Lactic Acid,Therapeut...
3	[30179656, 28641726, 28694230, 27796647, 28867...	[2019, 2017, 2018, 2017, 2017, 2017, 2015, 201...	[03, 12, 05, 02, 11, 05, 08, 06, 01, 08]	Intestines,Therapeutics,Catharsis,Lower Extrem...
4	[29787536, 24840763, 28273653, 26836795, 26409...	[2019, 2014, 2017, 2016, 2016, 2019, 2013, 201...	[05, 10, 11, 11, 05, 04, 03, 09, 02, 12]	Abdomen,Catheterization,Headache,Diuresis,Extr...
...
119	[28641726, 30179656, 28694230, 27796647, 28867...	[2017, 2019, 2018, 2017, 2017, 2017, 2017, 201...	[12, 03, 05, 02, 11, 05, 09, 09, 01, 03]	Extremities,Catharsis,Tazobactam,Abdomen,Oxyge...
120	[27128826, 30336861, 30226191, 29371130, 29587...	[2017, 2019, 2019, 2018, 2018, 2019, 2019, 201...	[04, 01, 09, 10, 08, 01, 06, 08, 11, 04]	Catharsis,Abdomen,Lung
121	[30595510, 21554494, 26465238, 26875969, 30056...	[2019, 2012, 2016, 2016, 2019, 2019, 2016, 201...	[03, 04, 09, 08, 07, 10, 10, 05, 10, 06]	Abdomen,Wounds and Injuries,Lung,Stroke,Aphasi...
122	[30081165, 30629460, 26220984, 25749853, 28545...	[2018, 2019, 2016, 2016, 2018, 2020, 2015, 201...	[12, 03, 06, 04, 12, 02, 02, 09, 05, 04]	Extremities,Catharsis,Thromboembolism,Foramen ...
123	[30662053, 29879068, 26849395, 31061178, 31223...	[2019, 2018, 2016, 2019, 2019, 2019, 2018, 201...	[02, 06, 12, 12, 07, 03, 12, 05, 04, 04]	Fever,Catharsis,Infections,Abdominal Pain,Abdo...

124 rows x 4 columns

Figura 4.7: *Flattening* del atributo respuesta.

Una vez aplanado los dos atributos, los uniremos en un único conjunto de datos, junto a los datos originales de la tabla *fed_hope_sugerencia* para poder trabajar con ellos. Esto es importante para mantener el id de cada observación, de cara a poder luego identificar el feedback de los profesionales sanitarios con cada observación.

4.3. Procesado de los datos

4.3.1. Análisis de los datos

Una vez tenemos los datos en formato tabular, observamos que existen ciertos atributos que contienen listas de opciones como son los atributos *pubmed_keys* (que corresponde a las palabras clave que la api de pubmed nos devuelve para esta observación), *articles* (que corresponde a los ids de los artículos relacionados con esa observación), *articlesRevisedYear* i *articlesRevisedMonth* (que corresponde a los años y meses de los artículos según están ordenados en el atributo *articles*)

Como nuestro **OP1** es poder recomendar artículos útiles, necesitamos tener una observación por artículo, para poder posteriormente analizar de manera independiente si ese artículo fue útil o no para la observación a la que hace referencia.

Por lo que necesitaremos expandir (duplicar) cada observación con solo un artículo que haga referencia a el. A continuación mostramos en la figura 4.8 el código para expandir el atributo *articles* (el resto de atributos su proceso sería similar).

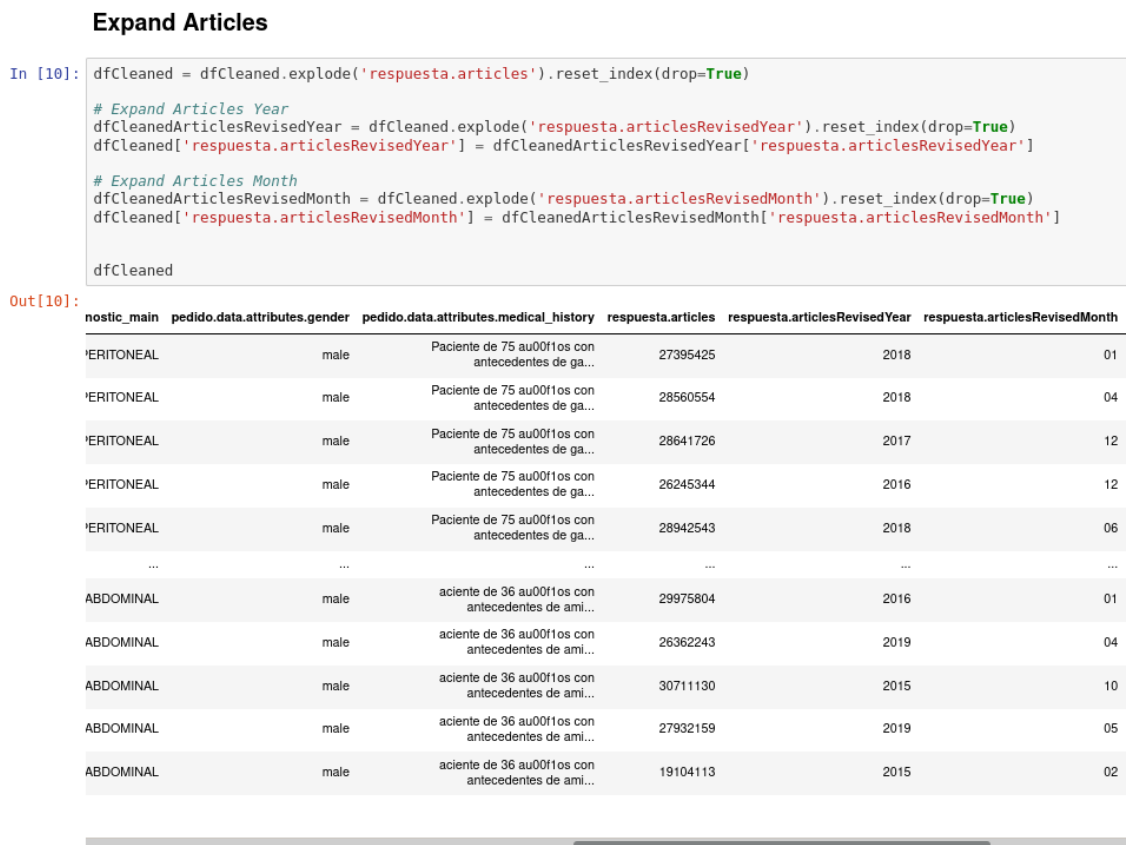


Figura 4.8: Expansión del atributo *articles*.

Después de tener un artículo por observación, observamos que tenemos atributos poco relevantes (como el atributo *data.type* o *Name* que contiene siempre el mismo valor) o que no contienen información alguna (como

el atributo *affected_organ*) como se puede observar en la figura 4.9. Eliminaremos estos atributos junto a otros con la misma casuística, para no generar ruido en el posterior análisis predictivo.

Out[11]:

	id	pedido.data.type	pedido.data.attributes.name	pedido.data.attributes.affected_organ	pedido.data.attributes.age	pedido.data.attributes.diagnostic_main
0	29	emr--em	None		75	FISTULA PERITONEAL
1	29	emr--em	None		75	FISTULA PERITONEAL
2	29	emr--em	None		75	FISTULA PERITONEAL
3	29	emr--em	None		75	FISTULA PERITONEAL
4	29	emr--em	None		75	FISTULA PERITONEAL
...
1235	152	emr--em	None		37	DOLOR ABDOMINAL
1236	152	emr--em	None		37	DOLOR ABDOMINAL
1237	152	emr--em	None		37	DOLOR ABDOMINAL
1238	152	emr--em	None		37	DOLOR ABDOMINAL
1239	152	emr--em	None		37	DOLOR ABDOMINAL

1240 rows x 13 columns

Figura 4.9: Se detectan algunos atributos con poca o nula relevancia.

Y con estos pasos hemos cubierto el [OS1](#)

4.3.2. Análisis de componentes principales

El análisis de componentes principales (o como se le conoce en inglés por '*Principal Component Analysis*' o *PCA*), es un componente fundamental en el análisis de los datos, ya que permite reducir el número de atributos de un conjunto de datos para eliminar el ruido que los posteriores análisis/modelos predictivos funcionen con mejor precisión.

Por poner un ejemplo sencillo, imaginemos que tenemos un conjunto de datos de modelos de coche con muchísimos atributos de estos, como por ejemplo, el nombre del modelo, el color, el número de puertas, la cilindrada y la potencia, entre otros. Probablemente si intentáramos analizar los datos en busca de cual es el modelo que menos consume y quisiéramos crear un modelo predictivo con este objetivo, podríamos observar a simple vista que tenemos atributos que no nos son necesarios y que generaría distracción (ruido) a la hora de conseguir nuestro objetivo, como es el caso del atributo color, o el nombre del modelo.

El *PCA* nos ayudara a encontrar cuales son los atributos más significativos del conjunto de datos para conseguir predecir el atributo que queremos, que en el caso que nos toca, es el atributo '*utilidad*'.

Es cierto que, en nuestro conjunto de datos, no tenemos un gran volumen de atributos, pero este proceso nos puede ayudar a eliminar atributos con poca relevancia, para así, poder simplificar el modelo predictivo final.

Para realizar el *PCA* nos ayudaremos de la librería *sklearn.decomposition* que ya nos ofrece implementada la lógica para ejecutarlo. Pero antes, tenemos que transformar todos los atributos de Categóricos (texto) a Continuos (números continuos). Este paso se realiza para que el *PCA* pueda realizar operaciones matemáticas sobre los valores de las observaciones. A este proceso se le conoce como factorización (*factorize*). Realizar esta transformación es tan sencillo como coger cada uno de los valores del atributo y asignarles un número. A continuación mostramos en la figura 4.10 un ejemplo de transformación de atributo categórico a continuo.

```
In [5]: dataDiagnosticMain, categoriesDiagnosticMain = pd.factorize(dfPCA['pedido.data.attributes.diagnostic_main'])
categoriesDiagnosticMain

Out[5]: Index(['FISTULA PERITONEAL', 'INSUFICIENCIA RESPIRATORIA', 'POLITRAUMATISMO',
              'ABDOMEN AGUDO', 'TORACOTOMIA', 'INFECCION DE PARTES BLANDAS',
              'DOLOR ABDOMINAL', 'INFECCION URINARIA', 'HEMORRAGIA DIGESTIVA',
              'ACV.ISQUEMICO', 'DISNEA', 'CETOACIDOSIS DIABETICA'],
             dtype='object')
```

0 => first element found => 'FISTULA PERITONEAL'

1 => second element found => 'INSUFICIENCIA RESPIRATORIA'

...

```
In [6]: dfPCA['pedido.data.attributes.diagnostic_main'] = dataDiagnosticMain
dfPCA.head(10)
```

```
Out[6]:
```

	pedido.data.attributes.age	pedido.data.attributes.diagnostic_main	pedido.data.attributes.gender	respuesta.pubmed_keys	articulo	uti
0	75	0	male	Abdomen,Adenocarcinoma,Antiemetics,Blood Cultu...	27395425	
1	75	0	male	Abdomen,Blood Culture,Catharsis,Diuresis,Drug ...	28694230	
2	36	1	male	Abdomen,Analgesics,Antitubercular Agents,Cipro...	28805236	
3	51	2	male	Abdomen,Analgesics,Bone,Catharsis,Electroconvu...	27537587	
4	51	2	male	Abdomen,Analgesics,Bone,Catharsis,Electroconvu...	28148670	
5	18	3	male	Abdomen,Anti-Bacterial Agents,Diuresis,Operati...	25055513	
6	18	3	male	Abdomen,Anti-Bacterial Agents,Diuresis,Operati...	29279563	
7	18	3	male	Abdomen,Anti-Bacterial Agents,Diuresis,Operati...	29279563	
8	18	3	male	Abdomen,Anti-Bacterial Agents,Diuresis,Operati...	28065368	
9	76	4	male	Abdomen,Amlodarone,Analgesia,Angiodysplasia,Hy...	30762794	

Figura 4.10: Factorización de un atributo.

Una vez realizada la factorización de todos los atributos necesarios, el siguiente paso, antes de ejecutar el PCA, será estandarizar todos los valores de las observaciones a un rango de entre 1 y -1. Esto se realiza para igualar la importancia de todos los atributos, ya que en el paso anterior, al factorizar los valores de los atributos, se nos puede dar el caso de tener valores muy altos (por ejemplo, al factorizar un atributo con 100 valores diferentes, se nos dará casos de observaciones que en un atributo tienen el valor 100, que puede ser más alto que otros valores que no se han transformado). Al realizar el PCA, si no se hace esta estandarización de los datos, los valores más altos se les dará más peso, pero no por eso pueden ser relevantes. Por lo que es imperativo el realizar esta estandarización.

Para realizar esta estandarización nos ayudaremos de la librería *sklearn.preprocessing* que ya tiene implementado el modelo de estandarización llamado *StandardScaler*. A continuación mostramos en la figura 4.11 un ejemplo de estandarización del conjunto de datos excluyendo de este el atributo a predecir (utilidad)

```
featuresTransformed = StandardScaler().fit_transform(x)
featuresTransformed
array([[ 0.91709628, -2.24479066,  0.          , -1.81819247, -0.19317962],
       [ 0.91709628, -2.24479066,  0.          , -1.65608091,  0.31397864],
       [-0.8747549 , -1.85670821,  0.          , -1.49396934,  0.35732434],
       [-0.18558137, -1.46862576,  0.          , -1.33185777, -0.13766811],
       [-0.18558137, -1.46862576,  0.          , -1.33185777,  0.100948  ],
       [-1.70176313, -1.0805433 ,  0.          , -1.16974621, -1.10687007],
       [-1.70176313, -1.0805433 ,  0.          , -1.16974621,  0.54253988],
       [-1.70176313, -1.0805433 ,  0.          , -1.16974621,  0.54253988],
       [-1.70176313, -1.0805433 ,  0.          , -1.16974621,  0.06842018],
       [ 0.96304118, -0.69246085,  0.          , -1.00763464,  1.12171293],
       [ 0.96304118, -0.69246085,  0.          , -1.00763464,  1.12171293],
       [ 0.96304118, -0.69246085,  0.          , -1.00763464,  0.66295943],
       [ 0.96304118, -0.69246085,  0.          , -1.00763464, -1.74384989],
       [-1.19636921, -0.30437839,  0.          , -0.84552307,  0.41640915],
       [-1.19636921, -0.30437839,  0.          , -0.84552307,  0.32427757],
       [ 1.10087588,  0.08370406,  0.          , -0.68341151, -0.59661724].
```

Figura 4.11: Estandarización del conjunto de datos.

Una vez realizado estos pasos podemos ejecutar el *PCA*. Con el conjunto de datos completo, y tal y como podemos ver en la figura 4.12, el *PCA* nos muestra que con solo 3 atributos, el modelo es capaz de explicar (predecir) el 95 % de las observaciones.

```
: print('explained variance ratio (first three components): %s' %
    str(pca.explained_variance_ratio_))
print('sum of explained variance (first three components): %s' %
    str(sum(pca.explained_variance_ratio_)))

explained variance ratio (first three components): [0.44726263 0.25669502 0.25009763]
sum of explained variance (first three components): 0.9540552760689917
```

Figura 4.12: Resultado del *PCA*.

Si extrapolamos los resultados de las observaciones a un gráfico que mostramos en la figura 4.13 dibujando los 3 componentes principales como si fueran 3 dimensiones, y pintamos los valores a predecir sobre estos (aplicando el color rojo a las observaciones que no consideran útiles, y azul a los que si se consideran útiles), podemos observar que la gran mayoría de los rojo se encuentran en el plano del componente principal 1 y dos, mientras que el azul esta mas repartidos en los dos planos.

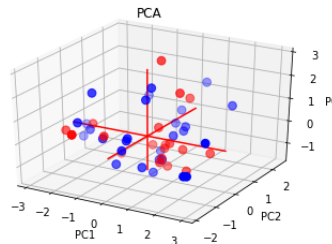


Figura 4.13: Proyección de los resultados del *PCA* en una gráfica.

Una vez ejecutado el modelo, podemos pedirle a este que nos indique cuales son los 3 componentes que ha detectado que son los principales. Para hacer esto le pediremos al modelo que nos muestre los resultados de los 3 componentes sobre los atributos del conjunto de datos. Estos resultados se pueden ver en la figura 4.14. Para saber cuales son los componentes, tendremos que quedarnos para cada componente principal, el valor del atributo que más se acerque a 1.

```
pd.DataFrame(pca.components_, columns=features, index = ['PC1', 'PC2', 'PC3'])
```

	pedido.data.attributes.age	pedido.data.attributes.diagnostic_main	pedido.data.attributes.gender	respuesta.pubmed_keys	articulo
PC1	-0.050066	0.705007	-1.110223e-16	0.705462	0.052744
PC2	0.760475	0.071805	-1.318390e-16	0.030411	-0.644668
PC3	-0.630542	-0.130930	-1.110223e-16	0.142297	-0.751682

Figura 4.14: Proyección de los resultados del *PCA* sobre los atributos del conjunto de datos.

Para este caso, el *PCA* nos muestra que los 3 componentes principales son, el atributo *pubmed_keys*, el atributo *age* y el atributo *diagnostic_main*.

Después de comentar los resultados con el cliente, acordamos hacer dos pruebas más:

- Ejecutar el *PCA* solo teniendo en cuenta las observaciones que tienen informado el atributo utilidad, y añadiendo el mes y año del artículo.

En este caso, el *PCA* nos muestra tal y como podemos ver en la figura 4.15, que necesitamos 5 atributos para que el modelo sea capaz de explicar (predecir) el 97 % de las observaciones.

```
print('explained variance ratio (first three components): %s' %
      str(pca.explained_variance_ratio_))
print('sum of explained variance (first three components): %s' %
      str(sum(pca.explained_variance_ratio_)))
```

explained variance ratio (first three components): [0.32833981 0.23094825 0.17265137 0.14192254 0.09557452]
sum of explained variance (first three components): 0.9694364891147347

Figura 4.15: Resultado del *PCA* de la segunda ejecución.

Para este caso, el *PCA* nos muestra como podemos ver en la figura 4.16, que los 5 componentes principales son, el atributo *diagnostic_main*, el atributo *Year*, el atributo *pubmed_keys*, el atributo *age* y el atributo *articulo*.

```
pd.DataFrame(pca.components_, columns=features, index = ['PC1', 'PC2', 'PC3', 'PC4', 'PC5'])
```

	pedido.data.attributes.age	pedido.data.attributes.diagnostic_main	pedido.data.attributes.gender	respuesta.articulosRevisedYear	respuesta.articulosRevisedMont
PC1	-0.094054	0.603912	-1.110223e-16	-0.365651	0.33837
PC2	0.135318	0.327845	-5.273559e-16	0.362710	-0.55606
PC3	-0.891259	-0.174252	-5.828671e-16	-0.236812	-0.13360
PC4	0.382620	-0.115174	-1.929013e-15	-0.621075	0.16245
PC5	0.098479	0.051453	3.774758e-15	-0.540677	-0.72925

PC1 => diagnostic_main PC2 => pubmed_keys/Year PC3 => pubmed_keys PC4 => age PC5 => articulo

Figura 4.16: Proyección de los resultados del *PCA* sobre los atributos del conjunto de datos para la segunda ejecución.

- Ejecutar el *PCA* solo teniendo en cuenta las observaciones que tienen informado el atributo utilidad, añadiendo el mes y año del artículo, eliminando los atributos *articulo* y *gender* y expandiendo el atributo *pubmed_keys* a un valor por observación, en vez de tener todas las palabras clave de cada artículo en una única observación.

En este caso, el *PCA* nos muestra tal y como podemos ver en la figura 4.17, que necesitamos 4 atributos para que el modelo sea capaz de explicar (predecir) el 90 % de las observaciones.

```
print('explained variance ratio (first three components): %s' %
      str(pca.explained_variance_ratio_))
print('sum of explained variance (first three components): %s' %
      str(sum(pca.explained_variance_ratio_)))
```

explained variance ratio (first three components): [0.31205207 0.24763151 0.19472659 0.14857077]
sum of explained variance (first three components): 0.9029809474497172

Figura 4.17: Resultado del *PCA* de la tercera ejecución.

Para este caso, el *PCA* nos muestra como podemos ver en la figura 4.18, que los 4 componentes principales son, el atributo *diagnostic_main*, el atributo *Month*, el atributo *pubmed_keys* y el atributo *age*.

```
pd.DataFrame(pca.components_, columns=features, index = ['PC1', 'PC2', 'PC3', 'PC4'])
```

	pedido.data.attributes.age	pedido.data.attributes.diagnostic_main	respuesta.articlesRevisedYear	respuesta.articlesRevisedMonth	respuesta.pubmed_keys
PC1	0.202186	0.688327	-0.236152	-0.080358	0.650462
PC2	-0.299028	-0.000864	-0.648047	0.698266	-0.055149
PC3	-0.908866	-0.006936	0.153411	-0.221720	0.318152
PC4	-0.015928	-0.145721	-0.702451	-0.673071	-0.179024

PC1 => diagnostic_main / pubmed_keys PC2 => articlesRevisedMonth PC3 => pubmed_keys PC4 => age

Figura 4.18: Proyección de los resultados del *PCA* sobre los atributos del conjunto de datos para la tercera ejecución.

4.4. Enriquecimiento de los datos. Aproximación por Vecinos más próximos (K-NN)

TODO

4.5. Modelos Predictivos

4.5.1. Regresión logística '*Logistic regression*'

TODO

4.5.2. Bosques Aleatorios '*Random Forest*'

TODO

4.5.3. Maquinas de Vector Soporte '*Support Vector Machines*'

TODO

4.6. Resultados

TODO

Capítulo 5

Conclusiones

TODO

Capítulo 6

Bibliográfica

Índice de figuras

4.1. Visualización de los atributos de la tabla <i>fed_hope_sugerencia</i>	6
4.2. Muestra de una observación del atributo pedido	6
4.3. Ejemplo de contenido del atributo respuesta de una observación	7
4.4. Visualización de los atributos de la tabla <i>fed_hope_sugerencia_feedback</i>	7
4.5. Lectura de los datos	8
4.6. <i>Flattening</i> del atributo pedido.	9
4.7. <i>Flattening</i> del atributo respuesta.	10
4.8. Expansión del atributo <i>articles</i>	12
4.9. Se detectan algunos atributos con poca o nula relevancia.	13
4.10. Factorización de un atributo.	14
4.11. Estandarización del conjunto de datos.	15
4.12. Resultado del <i>PCA</i>	15
4.13. Proyección de los resultados del <i>PCA</i> en una gráfica.	16
4.14. Proyección de los resultados del <i>PCA</i> sobre los atributos del conjunto de datos.	16
4.15. Resultado del <i>PCA</i> de la segunda ejecución.	17
4.16. Proyección de los resultados del <i>PCA</i> sobre los atributos del conjunto de datos para la segunda ejecución.	17
4.17. Resultado del <i>PCA</i> de la tercera ejecución.	18
4.18. Proyección de los resultados del <i>PCA</i> sobre los atributos del conjunto de datos para la tercera ejecución.	18

Capítulo 7

Anexos