Universitat Oberta de Catalunya (UOC)

Máster Universitario en Ciencia de Datos (*Data Science*)

TRABAJO FINAL DE MÁSTER

Área: 2

CLASIFICADOR DOCUMENTOS MÉDICOS HOPE

Autor: Rubén Vasallo Gonzalez

Profesor: Jordi Casas Roma

Tutor: Carlos Luis Sanchez Bocanegra

Co-Tutor: Rafael Pastor Vargas

Barcelona, 14 de diciembre de 2020

Créditos/Copyright

Esta obra está sujeta a una licencia de Reconocimiento - NoComercial - SinObraDerivada 3.0 España de CreativeCommons.

FICHA DEL TRABAJO FINAL

Título del trabajo:	CLASIFICADOR DOCUMENTOS MÉDICOS HOPE
Nombre del autor:	Rubén Vasallo Gonzalez
Nombre del colaborador/a docente:	Carlos Luis Sanchez Bocanegra, Rafael Pastor Vargas
Nombre del PRA:	Jordi Casas Roma
Fecha de entrega (mm/aaaa):	01/2021
Titulación o programa:	Máster Universitario en Ciencia de Datos
Área del Trabajo Final:	M2.979 - TFM
Idioma del trabajo:	Español
Palabras clave	hope, clasificador, medicina

Dedicatoria/Cita

Quiero dedicarle este trabajo a mis mentores *Carlos Luis Sanchez Bocanegra* y *Rafael Pastor Vargas* que me han ayudado, apoyado y guiado en todo momento para conseguir los objetivos del máster.

Agradecimientos

Quiero agradecer a *Carlos Luis Sanchez Bocanegra* por invitarme participar en el Proyecto HOPE y poder aportar mi granito de arena a este gran proyecto.

También quiero dar la gracias a todos los miembros del proyecto HOPE que me han dado la bienvenida al grupo y me han facilitado la vida en unas circunstancias en las que, cuando entre en este, no eran las más idóneas. La mayoría de integrantes del grupo son médicos y la saturación de trabajo que había por el COVID-19 era enorme. Tengo claro que sin ellos y sin la ayuda en especial de Carlos y *Nicolas Passadore* que ha estado luchando para conseguir un dataset con más observaciones, este Máster no habría sido posible.

Abstract

This Final Master's Thesis *FMT* was born from the need to be able to have in a simple, up-to-date and immediate way, medical bibliographic references cataloged according to the information and the patient's symptoms, being able to make a ranking of more or less interest function of the feedback provided by health professionals on these bibliographic references.

Resumen:

Este Trabajo de Final de Máster (*TFM*) nace de la necesidad de poder disponer de una manera sencilla, actualizada e inmediata, referencias bibliográficas médicas catalogadas según la información y los síntomas que tienen del paciente, pudiendo hacer una clasificación (*ranking*) de más o menos interés en función de la valoración (*feedback*) aportada por los profesionales sanitarios sobre estas referencias bibliográficas.

Palabras clave: clasificador, artículos, médicos, PCA, KNN, Regresión Logística, Random Forest, SVM

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

1.1. Descripción general del problema

El Trabajo de Final de Máster (*TFM*) que aquí se presenta nace de la necesidad por parte de los profesionales sanitarios de poder tener la información más exacta posible sobre las mejores referencias bibliográficas actuales sobre tratamientos a aplicar a un paciente, dado unos síntomas concretos. Actualmente existe infinidad de referencias medicas que los profesionales sanitarios pueden consultar, pero esta información es tan abundante que acaba siendo engorrosa de consultar[1]. Esto hace que, muchas veces sea complicado encontrar la información sobre estas referencias bibliográficas para tratar algunas enfermedades. 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.

Los profesionales sanitarios necesitan poder disponer de una plataforma que sea capaz de poder facilitarles estas referencias bibliográficas actuales, exactas y personalizadas al paciente, ajustándose a la búsqueda de la información que poseen, ya que ahora mismo, tienen que utilizar múltiples herramientas y buscadores para poder obtener esas referencias[2].

Bajo esa premisa nace el proyecto HOPE[3] (*Health Operations for Personalized Evidence*) con el objetivo de ayudar a estos profesionales sanitarios a encontrar estas referencias bibliográficas adaptadas y personalizadas al paciente. Actualmente existen bases de datos de confianza en donde los profesionales sanitarios y el publico en general puede buscar estas referencias de ensayos y estudios clínicos con información fiable y desarrollados con anterioridad, pero no siempre es fácil o rápido encontrar estos resultados[4].

El proyecto HOPE es un sistema basado en el procesado de lenguaje natural '*Natural Language Processing (NIP)*' para identificar la información de casos clínicos y referencias bibliográficas registradas en la Historia Clínica Electrónica, en base a los cuales realiza una búsqueda única por paciente para proporcionar al profesional sanitario recomendaciones de referencias bibliográficas donde constan tratamientos, estudios de investigación e información para ayudar al paciente. Todo en base a registros de fuentes científicas validas de información. En este proyecto, los profesionales sanitarios de todo el mundo puede consultar en una base de datos sanitaria estas referencias y ver que otros tratamientos relacionados con los síntomas de sus pacientes han dado resultado. Todo y con eso, el sistema no siempre devuelve las referencias actuales más relevantes por lo que, no siempre la información consultada es útil[5].

Ademas, actualmente 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 esta valoración (*feedback*), se pretende mejorar el sistema actual complementándolo con un modelo clasificador capaz de ayudar al actual a entregar realmente las referencias que son más útiles basándose en la valoración que los profesionales sanitarios dan al sistema.

Este Trabajo de final de máster (*TFM*) pretende ayudar al proyecto HOPE a mejorar su algoritmo de Inteligencia Artificial para que los resultados se ajusten a las necesidades de información que requieren los profesionales sanitarios, en base a las búsquedas personalizadas que puedan hacer respecto a la información que tienen de sus pacientes. Para hacer esto se realizara un estudio de aproximación a conocer cual es el mejor modelo predictivo que puede ayudar a devolver esa información lo más exacta, fiable y actualizada posible.

1.2. Motivación personal

En los duros tiempos en los que estamos viviendo actualmente, tanto económica como emocionalmente, es grato ver como la humanidad es capaz de dejar a un lado sus diferencias y unirse para afrontar problemas comunes. En el caso del proyecto HOPE, lo que más me atrajo fue la oportunidad de poder ayudar a encontrar soluciones a enfermedades que ya están en el último paso (o como médicamente se le describe, en cuidados paliativos).

Es evidente que este proyecto no va a dar una solución para curar cualquier enfermedad, pero si puede ayudar a los profesionales sanitarios a poder encontrar posibles soluciones a enfermedades complejas, ayudando a estos profesionales a buscar en la infinidad de documentación medica que existe, el tratamiento que más pueda ayudar a paliar o, quien sabe, curar una enfermedad

que ya se daba por incurable. Esto mismo es lo que me motiva y mucho el poder ayudar ante estas situaciones.

También me ha motivado muchísimo el conocer a gente profesional que, independientemente del país al que pertenece o la profesión que tiene, se una al proyecto HOPE para participar y ayudar con sus conocimientos a hacer de este mundo, un lugar un poco mejor en el que vivir. Esta experiencia me esta enriqueciendo muchísimo personalmente y espero poder seguir contribuyendo al proyecto cuando este máster acabe.

Capítulo 2

Objetivos del Máster

2.1. Objetivo principal

OP - Poder recomendar al profesional sanitario las referencias bibliográficas actuales útiles y personalizadas, que refuerce las tomas de decisiones en base a la información que se dispone de este, pudiendo realizar una clasificación (*ranking*) de más interés a menos.

2.2. 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 para poder seleccionar los atributos mas relevantes que influyen en los síntomas del paciente (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. 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*'.

OS7 - Comparar los resultados obtenidos de los 3 modelos.

Capítulo 3

Metodología

3.1. Análisis del contexto

Para poder comprender y abordar con éxito el OS1 se ha realizado dos reuniones en donde el cliente expuso 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, ya que muchas observaciones tenían información poco relevante que podía generar ruido[6].

3.2. Limitaciones detectadas

Realizando un primer análisis visual, se detecto que los datos aportados por el cliente eran insuficientes para completar el OP1, ya que solo se disponía de la información respecto de si una referencia bibliográfica había sido útil o no, pero no se disponía de la información suficientemente detallada con información del grado de utilidad que había tenido esa referencia para poder llegar a realizar una clasificación (*ranking*). El cliente nos comenta que en el momento actual no dispone de ese nivel de detalle.

Se acuerda con el cliente que intentara conseguir más volumen de información y con más detalle para poder realizar la clasificación (*ranking*). Se comenta que se realizara una aproximación para indicar si una referencia bibliográfica es valida, dejando para mas adelante la opción de poder realizar la clasificación (*ranking*) 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 planteo la posibilidad de, o intentar obtener más observaciones, o enriquecer las 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, conseguía obtener nuevas observaciones, estas serian añadidas al estudio para aproximar mejor la solución final.

3.3. Análisis de datos

Tras recibir los datos por parte del cliente y realizar el primer análisis superficial, se detecto que el volumen de datos no era suficiente como para que los resultados que pudieran surgir del estudio, fueran concluyentes. Todo y con eso se acordó (tal y como se refleja en el OS1) hacer una valoración de los datos para ver si se podrían enriquecer de algún modo, mientras el cliente intenta obtener más volumen de datos.

Para ello, transformamos los datos que se nos facilito desde una base de datos donde se encontraba la información guardada en formato json, a un formato columnar en el que poder aplicar los modelos predictivos. Una vez tenemos los datos transformados, 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[7] de pubmed nos devuelve para esta observación), *articles* (que corresponde a los ids de los artículos relacionados con esa observación), *articlesRevisedYear* y *articlesRevisedMonth* (que corresponde a los años y meses de los artículos según están ordenados en el atributo *articles*). Para poder recomendar artículos útiles, necesitamos tener una observación por artículo, para poder analizar posteriormente de manera independiente si ese artículo fue útil o no para la observación a la que hace referencia, por lo que aplicaremos las transformaciones necesarias para acabar obteniendo la representación de un artículo con su correspondiente *feedback* por parte de los profesionales sanitarios. Todos los pasos para realizar las transformaciones se pueden consultar en el anexo (Procesado de datos).

Aplicados las transformaciones anteriores, nos quedan los siguientes atributos que describimos a continuación:

• **pedido.data.attributes.age:** Nos indica la edad del paciente en formato numérico (ejemplos de valores => 75,86,40,...).

• **pedido.data.attributes.diagnostic_main:** Nos indica el diagnostico principal que dio el profesional sanitario para la enfermedad que tiene el paciente en formato texto (ejemplos de valores => *Fistula Peritoneal, Insuficiencia Respiratoria*,...).

• **pedido.data.attributes.gender:** Nos indica el sexo del paciente en formato texto (ejemplo de valor => *male*).

• **artículo:** Nos indica el identificador del artículo relacionado con el diagnostico principal en formato numérico (ejemplos de valores => 28694230,28805236,...).

• **respuesta.articlesRevisedYear:** Nos indica el año de revisión del artículo referenciado por el identificador del campo artículo en formato numérico (ejemplos de valores => 2018,2017,2016,...).

• **respuesta.articlesRevisedMonth:** Nos indica el mes de revisión del artículo referenciado por el identificador del campo artículo en formato numérico (ejemplos de valores => 4,12,6,9,...).

• **respuesta.pubmed_keys:** Nos indica las palabras clave relacionadas con el artículo del campo artículo en formato texto (ejemplos de valores => *'Abdomen, Adenocarcinoma, Antiemetics, Blood', 'Abdomen, Analgesics, Bone, Catharsis', 'Abdomen, Anti-Bacterial Agents, Diuresis'*,...).

• **utilidad:** Nos indica si un profesional sanitario ha considerado si el artículo referenciado en el campo artículo es útil para el diagnostico principal relacionado con la enfermedad del paciente. Este campo puede tener 3 valores: 1 para saber que el artículo es útil, 0 para indicar que no es útil y null para indicar que aun no se ha valorado.

3.4. Análisis de componentes principales

El análisis de componentes principales (o como se le conoce en ingles 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[6] que los posteriores análisis/modelos predictivos funcionen con mejor precisión[8].

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 modelos 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[6]) 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[8], 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. Ademas realizaremos la transformación de 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*).

También estandarizaremos los valores 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 realizar la factorización, 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 el PCA se han seguido 3 estrategias, para valorar que impacto tiene las observaciones según el conjunto de datos a analizar. Hemos realizado 3 iteraciones con el siguiente conjunto de datos:

- **1:** El dataset completo con las transformaciones mencionadas en el apartado Análisis de datos

- **2:** El mismo dataset de la anterior iteración pero cogiendo solo las observaciones que se ha informado el atributo utilidad. Se decide realizar esta prueba para eliminar las observacio-

nes que no tienen informado ese atributo para eliminar el ruido[6] que provoca tener muchas observaciones con un valor nulo.

• **3:** El mismo dataset realizado en la iteración 2 pero añadiendo el mes y año del artículo, eliminando los atributos *gender* (ya que todas las observaciones tienen el mismo valor) y artículo (que solo representa un identificador de un artículo). Ademas se expande el atributo *respuesta.pubmed_keys* para que exista una observación por cada *keyword*. Se decide añadir el mes y año de la revisión del articulo porque puede añadir valor. Ademas expandimos las *keywords* a una por observación ya que puede ser relevante analizar las *keywords* de manera individual en vez de en conjunto por cada observación.

Una vez realizado esa transformación ya podemos ejecutar el PCA. Todos los pasos para realizar el PCA se pueden consultar en el anexo (Iteración 1) para la iteración 1, el anexo (Iteración 2) para la iteración 2 y el anexo (Iteración 3) para la iteración 3.

3.5. Enriquecimiento de los datos

Debido a que tenemos un conjunto de datos con poco volumen, se acuerda el intentar enriquecer los datos (OS3) que no tienen un valor '*utilidad*' definido. Como vimos en el apartado de análisis de datos, en el conjunto de datos aparecen muchas observaciones que no tienen indicado el atributo '*utilidad*', ya que todavía no han sido valorados por los profesionales sanitarios. Estas observaciones se podrían rellenar intentando aproximarlas teniendo en cuenta valores de observaciones con similares características. En los siguientes apartados valoraremos si el resultado de esas aproximaciones seria lo suficientemente correcto para acabar aplicándolo.

Para realizar esta operación utilizaremos el algoritmo de Enrquicecimiento por Aproximación de Vecinos más próximos (conocido por K-Nearest-Neighbor o K-NN). Este Algoritmo pretende asociar si un registro pertenece a un conjunto de datos o a otro dependiendo de la aproximación de otros resultados de los que si se conoce su valor [9].

Para poder ejecutar el algoritmo de K-NN necesitamos realizar la transformación de todos los atributos Categóricos (texto) a Continuos (números continuos) y estandarizar, igual que se realizo para el caso del PCA. Esto es debido a que el algoritmo necesita tener valores numéricos para poder trabajar con los datos[10]. Con esta transformación ya podemos realizar operaciones matemáticas con el conjunto de datos.

Para realizar el K-NN se han seguido 2 estrategias, para valorar que impacto tiene las observaciones según el conjunto de datos a analizar. Hemos realizado 2 iteraciones con el siguiente conjunto de datos:

- **1:** El dataset completo con las transformaciones mencionadas en el apartado Análisis de datos eliminando las observaciones que no tenían identificado el atributo edad.

- **2:** El mismo dataset realizado en la iteración 1 pero eliminando los atributos *gender* (ya que todas las observaciones tienen el mismo valor) y artículo (que solo representa un identificador de un artículo). Ademas se expande el atributo *respuesta.pubmed_keys* para que exista una observación por cada keyword.

Una vez realizado esa transformación ya podemos ejecutar el K-NN. Todos los pasos para realizar el K-NN se pueden consultar en el anexo (Iteración 1) para la iteración 1 y el anexo (Iteración 2) para la iteración 2.

3.6. Regresión Logística

Una vez analizado el conjunto de datos aportado por el cliente, podemos ver que finalmente el atributo a predecir *utilidad* es de tipo booleano (0 haciendo referencia a los documentos con poco o nada de interés para los profesionales sanitarios y 1 para los que si son de interés).

Como el problema a resolver es de clasificación, escogemos como posible modelo de aprendizaje supervisado para clasificación a analizar el conocido como Regresión Logística '*Logistic regression*'[11] OS4. Escogemos este como posible modelo de confianza a analizar debido a que es un modelo muy sencillo de entrenar y comprender su comportamiento[12].

Igual que para el caso del K-NN, realizaremos la transformación de todos los atributos Categóricos (texto) a Continuos (números continuos) y estandarizaremos los valores.

Para ejecutar el modelo de Regresión Logística se han seguido 2 estrategias, para valorar que impacto tiene las observaciones según el conjunto de datos a analizar. Hemos realizado 2 iteraciones con el siguiente conjunto de datos:

- **1:** El dataset completo con las transformaciones mencionadas en el apartado Análisis de datos eliminando las observaciones que no tenían identificado el atributo edad y escogiendo los atributos que nos ha indicado como relevantes el caso de estudio 2 del PCA.

• **2:** El mismo dataset realizado en la iteración 1 pero eliminando los atributos *gender* (ya que todas las observaciones tienen el mismo valor) y artículo (que solo representa un identificador de un artículo). Ademas se expande el atributo *respuesta.pubmed_keys* para que exista una observación por cada keyword. También escogemos solo los atributos que nos ha indicado como relevantes el caso de estudio 3 del PCA.

Todos los pasos para realizar la Regresión Logística se pueden consultar en el anexo (Iteración 1) para la iteración 1 y el anexo (Iteración 2) para la iteración 2.

3.7. Bosques Aleatorios '*Random Forests*'

Otro modelo que escogemos como posible para intentar resolver el problema, dada la casuistica del dataset y el atributo a predecir, es el modelo de aprendizaje supervisado para clasificación conocido como Bosques Aleatorios '*Random Forests*'[13] OS5. Escogemos este como posible modelo de confianza a analizar, debido a que su algoritmo suele tener un porcentaje de acierto elevado en conjuntos de datos con muchos atributos.

Igual que para el caso del K-NN, realizaremos la transformación de todos los atributos Categóricos (texto) a Continuos (números continuos) y estandarizaremos los valores (**Nota**: Este modelo no es necesario estandarizar los valores pero he decidido hacerlo para igualar los valores de los atributos de entrada para todos los modelos).

Para ejecutar el modelo de Bosques Aleatorios se han seguido las mismas 2 estrategias que para el caso del modelo de Regresión Logística, para así poder comparar los resultados de este modelo con el anterior.

Todos los pasos para realizar los Bosques Aleatorios se pueden consultar en el anexo (Iteración 1) para la iteración 1 y el anexo (Iteración 2) para la iteración 2.

3.8. Máquinas de vector soporte '*Support Vector Machines*'

Finalmente estudiaremos un tercer modelo como posible para intentar resolver el problema, el conocido como Máquinas de Vector Soporte '*Support Vector Machines*'[14] OS6. Escogemos este como posible modelo de confianza a analizar, debido a su facilidad de entrenamiento, al tener multitud de funciones kernel[15] (que se van ampliando[16] con el paso del tiempo) para

permitir encontrar un ajuste optimo del modelo. Por contra, este punto fuerte puede también convertirse en su gran inconveniente debido a la complejidad de algunas funciones kernel a la hora de interpretar y comprender su comportamiento. Todo y con eso merece la pena analizarlo para ver si puede convertirse en un posible candidato para solucionar el problema descrito por el cliente.

Igual que para el caso del K-NN, realizaremos la transformación de todos los atributos Categóricos (texto) a Continuos (números continuos) y estandarizaremos los valores.

Para ejecutar el modelo de Máquinas de vector soporte se han seguido las mismas 2 estrategias que para el caso del modelo de Regresión Logística, para así poder comparar los resultados de este modelo con los anteriores.

Todos los pasos para ejecutar el modelo de Máquinas de Vector Soporte se pueden consultar en el anexo (Iteración 1) para la iteración 1 y el anexo (Iteración 2) para la iteración 2.

Capítulo 4

Planificación

4.1. Fechas importantes

A continuación se detalla en la tabla 4.1 las fechas clave del proyecto.

Nombre	Fecha de inicio	Fecha de fin	Duración
Reunión inicial con el cliente	22/09/20	22/09/20	1
Redacción de objetivos	23/09/20	27/09/20	5
Análisis de mercado	28/09/20	18/10/20	21
Enriquecer el DataSet	19/10/20	01/11/20	14
Diseño del Modelo 1	02/11/20	15/11/20	14
Diseño del Modelo 2	16/11/20	29/11/20	14
Diseño del Modelo 3	30/11/20	13/12/20	14
Redacción de conclusiones	14/12/20	27/12/20	14
Preparación de la Defensa	28/12/20	01/01/21	5

Cuadro 4.1: *Tabla de fechas clave del proyecto.*

4.2. Diagrama Gantt

A continuación se muestra en la figura 4.1 el diagrama Gantt del proyecto.

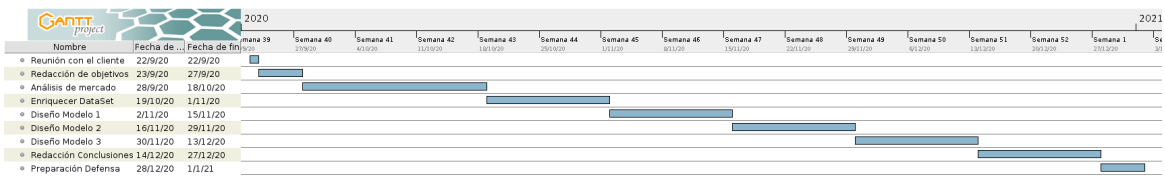


Figura 4.1: Gantt del proyecto.

Capítulo 5

Estado del arte

5.1. Recomendadores actuales

A día de hoy, existen multitud de clasificadores y recomendadores[17] que se usan a diario para realizar todo tipo de recomendaciones, que van desde cual es el mejor producto en base a unas necesidades, pasando por clasificar piezas o productos en una fabrica para organizar la disponibilidad de estas, hasta incluso a recomendadores de canciones, series o películas en base a unas características de quien solicita estas.

Esto se ha conseguido gracias a la investigación y al IOT (Internet de las cosas 'Internet of Things') que facilita la vida a los investigadores y sobre todo a las personas. Antiguamente eran los usuarios los que, si querían comparar algo, tenían que buscar y buscar por tiendas para encontrar, o bien el mismo producto a distintos precios o bien productos alternativos al que quería adquirir o que directamente desconocía de la existencia de esa alternativa. Esto le consumía muchas horas (o incluso días) y ademas no tenia la garantía de haber abarcado todas las posibilidades[18].

Hoy en día, a través de Internet y gracias a los Recomendadores/Clasificadores actuales, el usuario puede consultar todas las alternativas y precios y estar mas seguro de que hace una buena elección antes de hacer la inversión[19].

Los sistemas de recomendación/clasificación existen desde hace muchos años[20]. Han estado ahí para ayudarnos sin darnos cuenta, aunque al principio fueron muy rudimentarios. A medida que las características y el volumen de opciones crecía, se hacia mas complicado y lento realizar esa clasificación/recomendación. Y en ese punto es donde entro los algoritmos de aprendizaje

automático (*Machine Learning*) para ayudar a que esas clasificaciones y recomendaciones fueran posibles en un tiempo razonable.

Dos empresas que quizás merezcan mención especial (gracias a la fama que consiguieron al implementar estos algoritmos) son *Spotify*[21] con su recomendador de canciones y *Netflix*[22] con su recomendador de series. Ambos utilizan el poder de los algoritmos de recomendadores de *Machine Learning* para analizar el comportamiento que tienen sus usuarios en las correspondientes plataformas y sugerir recomendaciones basándose en estos comportamientos.

5.1.1. Recomendadores en el ámbito de la medicina

Pese a que en el ámbito de la medicina se ha utilizado mucho la estadística, el uso de estos recomendadores ha sido mas discreto, o al menos no se ha dado a conocer su uso al publico potencial, llegando a quedarse en la fase de caso de estudio sin llegar a convertirse en un producto final. Un ejemplo de esto es el recomendador de medicamentos personalizado[23] que se basa en el historial clínico del paciente para recomendar la medicación que se ha de tomar este. Este es un caso de uso muy similar a nuestro OP1, aunque en nuestro caso, buscamos recomendar referencias bibliográficas clínicas.

También se conocen casos donde se han usado estos Clasificadores/Recomendadores para clasificar muestras o grupos de individuos para realizar un posterior análisis del grupo, como por ejemplo este caso de uso[24] que utiliza un clasificador para clasificar plaquetas para un posterior ensayo medico.

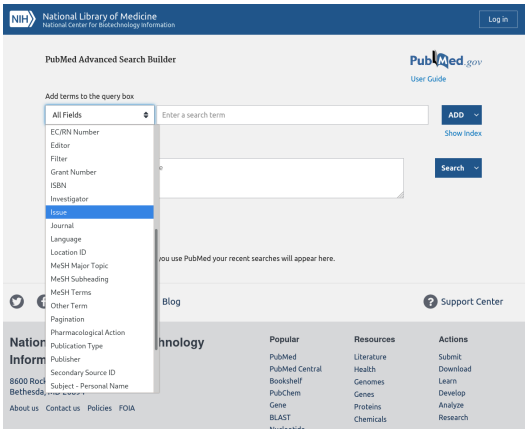
Todo y con eso, como comentamos anteriormente, el uso de Clasificadores/Recomendadores en este ámbito suele pasar a estar en segundo plano por lo que se ha de intuir su uso para darse cuenta que sin ellos, muchos estudios no serian posibles a día de hoy.

5.1.2. Recomendadores en el ámbito de la salud

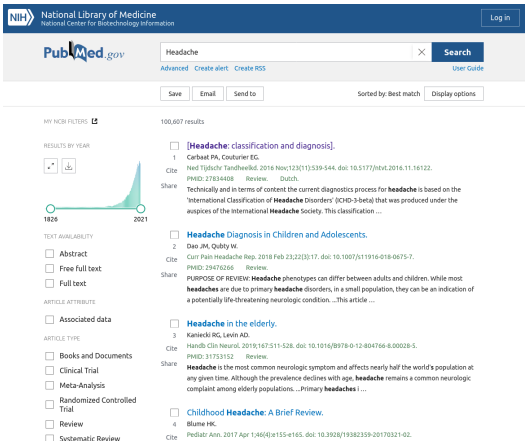
Al igual que en el caso anterior, el uso de recomendadores en el ámbito de la salud pasa desapercibido y es raro la vez que se destaque como solución final para la ayuda a los profesionales sanitarios.

A continuación destacamos los más conocidos en este ámbito:

• **Pubmed[25]:** Esta web Creada por el Centro Nacional de Biotecnología *National Center for Biotechnology Information (NCBI)* Permite hacer búsquedas por texto libre y permite realizar filtros por año, tipo de artículo y disponibilidad del texto. En el listado también se puede observar el estado de los artículos, si la publicación es definitiva o aun se esta revisando su contenido. El orden de los resultados es por aproximación al texto que se busque (mejor acierto), aunque permite modificar la ordenación por fecha del artículo, nombre de autor u origen del artículo. En las figuras 5.1a y 5.1b puede ver el aspecto de la pagina y los resultados de una búsqueda.



(a) Vista principal de Pubmed.



(b) Resultados de búsqueda de Pubmed.

Figura 5.1: Web de Pubmed.

• **ClinicalTrials**[26]**:** Esta web es una base de datos de estudios clínicos financiados con fondos públicos y privados realizados en todo el mundo. Permite realizar búsquedas y filtrados complejos de dolencias, mostrando los resultados en formato lista ordenados por fecha mas reciente, aunque el usuario puede elegir otras ordenaciones y filtrados. El listado también muestra el estado del documento (si ya esta finalizada la investigación o aun esta en proceso) y otra información relevante como el tratamiento que se esta aplicando. En las figuras 5.2 y 5.3 puede ver el aspecto de la pagina y los resultados de una búsqueda.

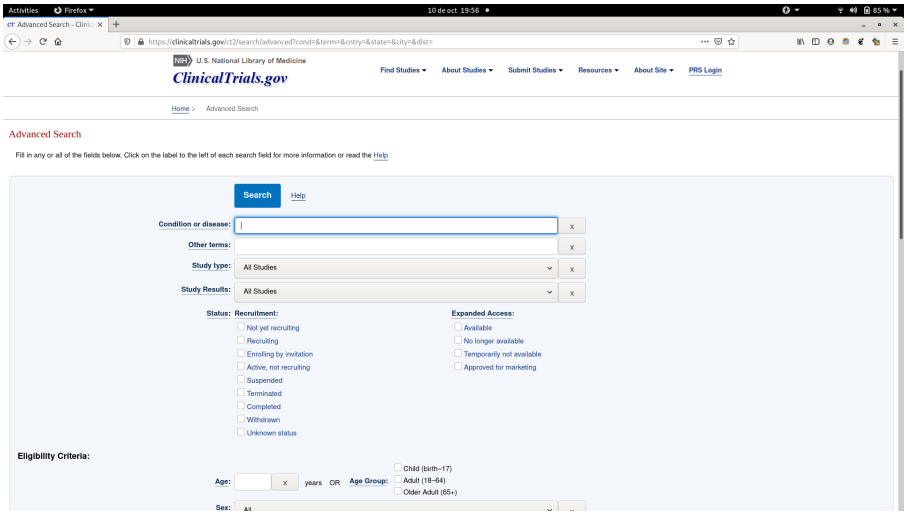


Figura 5.2: Vista principal de ClinicalTrials.

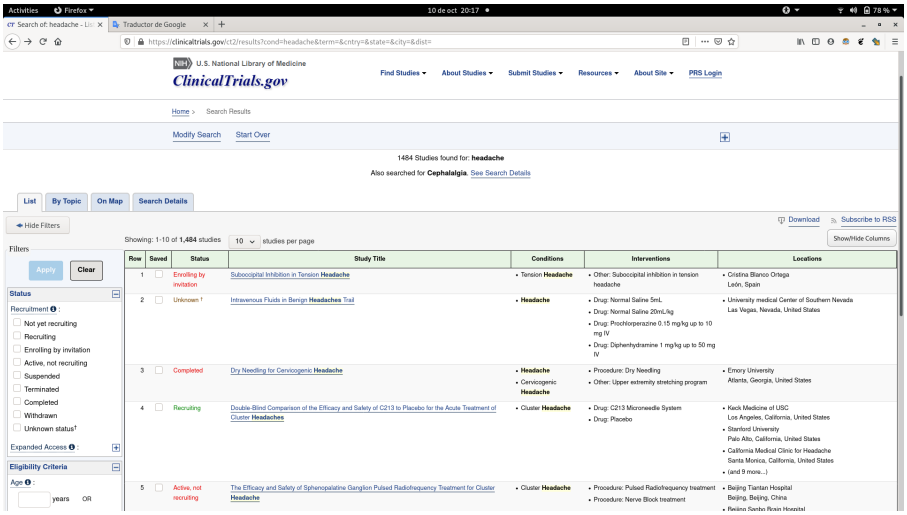


Figura 5.3: Resultados de búsqueda de ClinicalTrials.

• **MedlinePlus[27]:** Esta web creada por Biblioteca Nacional de Medicina de EE. UU. (NLM, por sus siglas en inglés), esta más orientada al paciente que las anteriores, y permite buscar artículos médicos basándose en texto relacionado. Por ejemplo si ponemos un tipo de síntoma en el buscador, este nos muestra artículos relacionados con ese texto como si de un buscador de Internet común se tratara, aunque a diferencia de estos, MedlinePlus promete brindar información de calidad y relevante de salud y bienestar que sea confiable y fácil de entender. Aparentemente este buscador muestra los resultados en orden de relevancia aunque, igual que en los casos anteriores, no permite valorar si esos artículos son realmente interesantes para la búsqueda relacionada.En las figuras 5.4a y 5.4b puede ver el aspecto de la pagina y los resultados de una búsqueda.

(a) Vista principal de MedlinePlus.

(b) Resultados de búsqueda de MedlinePlus.

Figura 5.4: Web de MedlinePlus.

5.2. Recomendadores en el futuro

Los recomendadores/clasificadores han ayudado mucho en la medicina, sobre todo en el ámbito de la investigación, en donde se requiere un trabajo previo para poder tener un conjunto de datos valido para el posterior estudio que se quiera realizar. En el caso que nos toca, ayudará a los profesionales sanitarios a poder encontrar otras referencias bibliográficas personalizadas y adaptadas al estado del paciente, para así poder valorar alternativas. Este trabajo seria un trabajo eterno de realizar, por no decir casi imposible sin la ayuda de estos recomendadores/clasificadores, debido al gran volumen de información que cada día se genera. Estos algoritmos de recomendación/clasificación, no solo seguirán siendo un pilar básico en el ámbito medico, sino que seguramente aparecerán nuevos algoritmos de clasificación/recomendación que facilitaran las futuras investigaciones, como ya se esta observando en los nuevos recomendadores para el control de la glucosa en la sangre[28].

Capítulo 6

Resultados

6.1. Análisis de componentes principales

6.1.1. Conjunto de datos completo

Para este caso, al ejecutar el *PCA* con el conjunto de datos completo, este nos muestra tal y como podemos ver en la figura 6.1, 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 6.1: Resultado del *PCA*.

Dado que el *PCA* solo nos muestra 3 atributos como relevantes, podemos representar los resultados de las observaciones a un gráfico que mostramos en la figura 6.2 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.

Si le pedimos al modelo que nos indique cuales son los 3 componentes que ha detectado que son los principales, este nos muestra los resultados que se pueden ver en la figura 6.3. Para saber cuales son los componentes, tendremos que quedarnos para cada componente principal, el valor del atributo que más se acerque a 1.

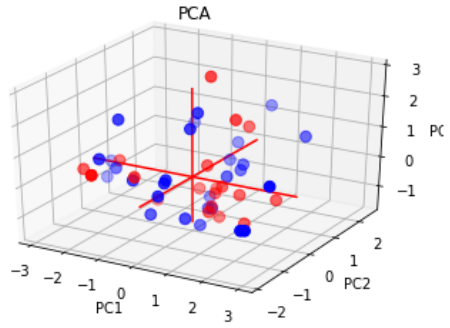


Figura 6.2: Proyección delos resultados del *PCA* en una gráfica.

```
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 6.3: 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***.

6.1.2. Conjunto de datos solo con atributo utilidad definido

En este caso, el *PCA* nos muestra tal y como podemos ver en la figura 6.4, 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 6.4: Resultado del *PCA* de la segunda ejecución.

Para este caso, el *PCA* nos muestra tal y como podemos ver en la figura 6.5, que los **5 componentes principales** son, **el atributo *diagnostic_main*, el atributo *Year*, el atributo *pubmed_keys*, el atributo *age* y el atributo *artículo***.

```
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.articlesRevisedYear	respuesta.articlesRevisedMont
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 6.5: Proyección de los resultados del *PCA* sobre los atributos del conjunto de datos para la segunda ejecución.

6.1.3. Conjunto de datos solo con atributo utilidad definido, añadiendo el mes y año del artículo, eliminando los atributos *gender* y artículo y expandiendo el atributo *respuesta.pubmed_keys*

En este caso, el *PCA* nos muestra tal y como podemos ver en la figura 6.6, 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 6.6: Resultado del *PCA* de la tercera ejecución.

Para este caso, el *PCA* nos muestra como podemos ver en la figura 6.7 , 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 6.7: Proyección de los resultados del *PCA* sobre los atributos del conjunto de datos para la tercera ejecución.

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

6.2.1. Características de los conjuntos de datos a analizar

Una vez realizado todas las transformaciones necesarias para poder ejecutar el algoritmo de *K-NN* podemos observar que tanto el '*Conjunto de datos solo con atributo utilidad definido*' (Caso de estudio 1) como el '*Conjunto de datos sólo con atributo utilidad definido, añadiendo el mes y año del artículo, eliminando los atributos gender y artículo y expandiendo el atributo respuesta.pubmed_keys*' (Caso de estudio 2) **se observa el atributo a predecir (utilidad) sesgado**[29] (no tienen un balanceo de datos equilibrado en el atributo a predecir), teniendo más resultados con valor 1 que con valor 0. Esto puede suponer que el modelo resultante tienda a estar sesgado[29] hacia los resultados con valor 1 (debido a que tiene más datos en los que basarse para dar el resultado 1 frente a los otros datos como podemos ver en la figura 6.8.

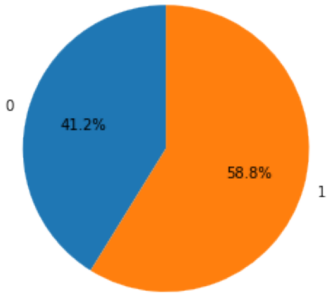


Figura 6.8: Distribución del atributo a predecir (utilidad) en el conjunto de entrenamiento.

También, para el **caso de estudio 2**, se da el caso que la distribución del **atributo *pubmed_keys***, una vez expandido, también **se observa sesgado** mostrando algunos resultados con pocas *keywords* y otros como '*Diuresis*' o '*Abdomen*' con muchas observaciones como podemos ver en la figura 6.9. Esto puede condicionar el comportamiento del modelo final igual que en el caso anterior, dando sospechas a que el modelo podría esta sobre ajustado[30].

Finalmente, **para el entrenamiento y posterior validación** de los modelos nos ayudaremos de la función '*train_test_split*'[31] del paquete '*sklearn.model_selection*' que nos permite **dividir el conjunto de datos en dos grupos**, siendo el **primer grupo** para realizar el **entrenamiento** de los modelos que **corresponde al 75 % del total** de observaciones y el segundo grupo para realizar la **validación** de los modelos que **corresponde al 25 % del total** de observaciones.

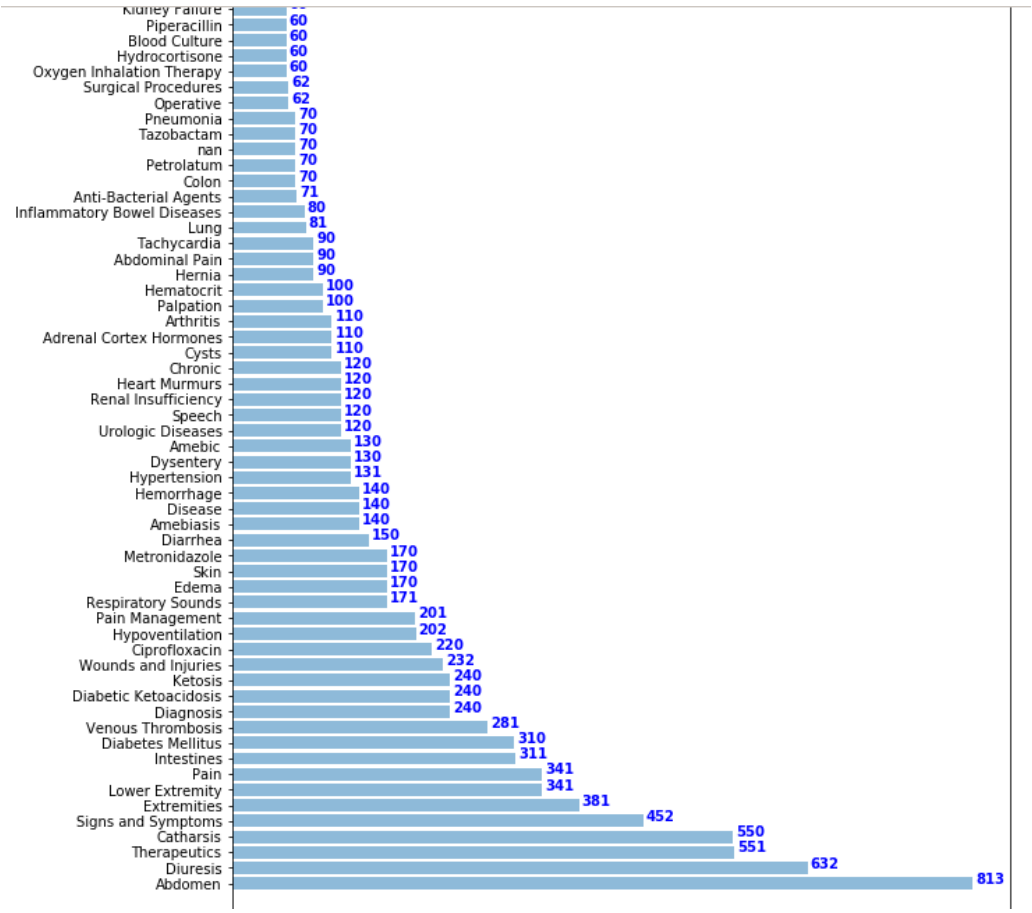


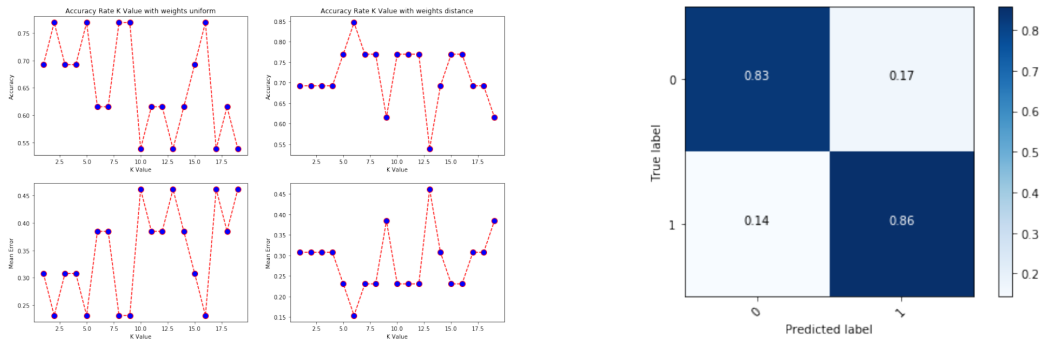
Figura 6.9: Distribución del atributo *pubmed_keys* en el conjunto de entrenamiento.

6.2.2. (Caso de estudio 1) Resultados del entrenamiento con el Conjunto de datos solo con atributo utilidad definido

Antes de poder entrenar el algoritmo *K-NN* es necesario realizar un estudio para detectar a partir de que número de vecinos mas próximos (valor K), podemos dar por supuesto que el resultado esta relacionado con ellos. El algoritmo también nos permite poder decidir como calcular la distancia entre los vecinos (bien dando más relevancia a los vecinos con menos distancia '*distance*'[32] o bien dando la misma relevancia a todos los vecinos que están próximos '*uniform*'[32]).

Para realizar este estudio ejecutaremos el modelo con diferentes valores de K y contrastaremos que porcentaje de acierto tiene el modelo con un subconjunto del propio conjunto de datos que conocemos su valor final. Como podemos ver en la figura 6.10a, el valor más optimo de **K es 6** utilizando el calculo de la distancia **'*distance*'**, con un **porcentaje de acierto del 85 %**.

Si mostramos la matriz de confusión[33] (figura 6.10b) para poder ver que porcentaje de aciertos tiene el modelo con el conjunto de test, vemos que tiene un porcentaje de aciertos bastante aceptable pese a tener más acierto para los casos con resultado 1 vrs. los resultados con valor 0 como comentamos anteriormente.



(a) Calculo del valor K para el caso de estudio 1.

(b) Matriz de confusión para el modelo K-NN en el caso de estudio 1.

Figura 6.10: Modelo K-NN en el caso de estudio 1.

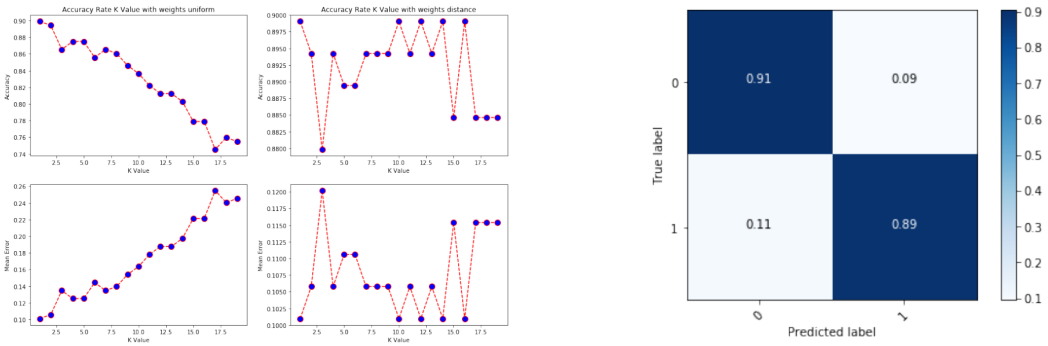
6.2.3. (Caso de estudio 2) Conjunto de datos solo con atributo utilidad definido, añadiendo el mes y año del artículo, eliminando los atributos *gender* y artículo y expandiendo el atributo *respuesta.pubmed_keys*

Igual que en el caso anterior, realizaremos el estudio para detectar a partir de que número de vecinos mas próximos (valor K), podemos dar por supuesto que el resultado esta relacionado con ellos. A diferencia del caso anterior, podemos ver en la figura 6.11a, que el valor más optimo de **K es 1** utilizando el calculo de la distancia **'*uniform*'**, con un **porcentaje de acierto del 90 %**.

Si mostramos la matriz de confusión[33] (figura 6.11b) para poder ver que porcentaje de aciertos tiene el modelo con el conjunto de test, vemos que tiene un porcentaje de aciertos bastante elevado por lo que podemos afirmar que **el modelo es sospechoso de estar sobre ajustado**[30].

6.2.4. Conclusiones del algoritmo K-NN

Pese a que aparentemente, el caso de estudio 1 tiene unos resultados aceptables, **aconsejamos** al cliente **no utilizar el algoritmo de *K-NN*** para enriquecer los datos del dataset,



(a) Calculo del valor K para el caso de estudio 2.

(b) Matriz de confusión para el modelo K-NN en el caso de estudio 2.

Figura 6.11: Modelo K-NN en el caso de estudio 2.

prediciendo las observaciones que se desconocen su resultado final. Esto es **debido** a que detectamos que **el conjunto de datos se encuentra sesgado** en varios atributos, lo que hará que el modelo tienda a dar falsos positivos (como hemos podido ver en el caso de estudio 2), perjudicando el entrenamiento de posteriores modelos predictivos que estudiaremos en este documento.

6.3. Regresión Logística

6.3.1. Características de los conjuntos de datos a analizar

Al realizar las mismas transformaciones que para el caso de K-NN, las características de los conjuntos de datos son las mismas.

6.3.2. (Caso de estudio 1) Resultados del entrenamiento con el Conjunto de datos solo con atributo utilidad definido y escogiendo los atributos que nos ha indicado como relevantes el caso de estudio 2 del PCA

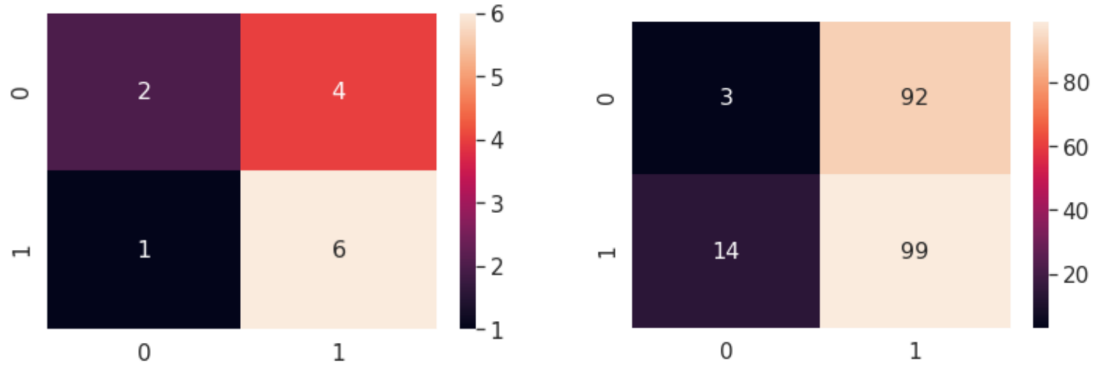
Realizamos el entrenamiento del modelo de Regresión Logística[11] y aplicamos una validación cruzada[34] para valorar su precisión. Este nos muestra una **precisión de cerca del 51 % con una desviación estándar de casi el 28 %**, lo que nos indica que el modelo tiene una precisión baja. Al realizar la predicción **sobre el conjunto de validación**, este nos da **una precisión del 65 %**. Lo que nos confirma que el modelo **tiene una precisión muy baja** (llegando casi al nivel de la aleatoriedad).

Si mostramos la matriz de confusión[33] (figura 6.12a) para poder ver que porcentaje de aciertos tiene el modelo con el conjunto de test, confirmamos que el porcentaje de aciertos es bajo.

6.3.3. (Caso de estudio 2) Conjunto de datos solo con atributo utilidad definido, añadiendo el mes y año del artículo, eliminando los atributos *gender* y artículo y expandiendo el atributo *respuesta.pubmed_keys*. También escogemos solo los atributos que nos ha indicado como relevantes el caso de estudio 3 del PCA

Igual que para el caso anterior realizamos el entrenamiento y aplicamos la validación cruzada para valorar su precisión. Este nos muestra **una precisión aun menor que para el caso de estudio 1 (49 %)**.

Si mostramos la matriz de confusión[33] (figura 6.12b) para poder ver que porcentaje de aciertos tiene el modelo con el conjunto de test, podemos ver con suficiente claridad que el **porcentaje de aciertos es muy bajo**.



(a) Matriz de confusión para el modelo Regresión Logística en el caso de estudio 1.

(b) Matriz de confusión para el modelo Regresión Logística en el caso de estudio 2.

Figura 6.12: Matriz de confusión para el modelo Regresión Logística.

6.3.4. Conclusiones del modelo de Regresión Logística

Después de realizar el entrenamiento y posterior validación de los modelos podemos **desaconsejar su uso debido a su bajo porcentaje de aciertos**. Es posible que con un volumen de

datos superior este se pueda comportar mejor, pero no se da el caso con el conjunto de datos que tenemos en la actualidad.

6.4. Bosques Aleatorios '*Random Forests*'

6.4.1. Características de los conjuntos de datos a analizar

Al realizar las mismas transformaciones que para el caso de K-NN, las características de los conjuntos de datos son las mismas.

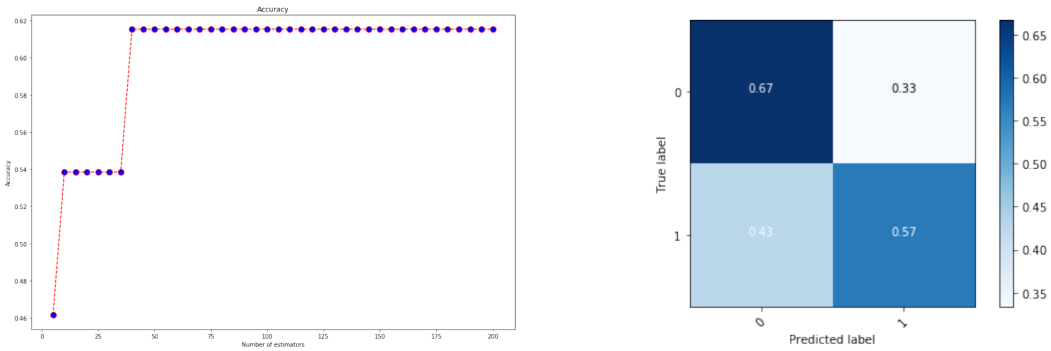
6.4.2. (Caso de estudio 1) Resultados del entrenamiento con el Conjunto de datos solo con atributo utilidad definido y escogiendo los atributos que nos ha indicado como relevantes el caso de estudio 2 del PCA

Para realizar el entrenamiento y posterior validación del modelo de Bosques Aleatorios '*Random Forests*' es necesario detectar antes a partir de que número de arboles podemos considerar que estos dan el valor a predecir correcto. Es lógico pensar que cuantos más arboles refuercen la decisión, más acertada sera esta pero, lo que hay que medir es, hasta que punto merece la pena seguir 'preguntando' si un registro es clasificado en un grupo u otro.

Esto es debido a que el modelo de Bosques Aleatorios '*Random Forests*'[13] genera versiones diferentes del conjunto de entrenamiento usando muestreo con reemplazo, usando la estrategia de *bagging*[35]. Esto significa que, durante el proceso de construcción de cada árbol de decisión, se selecciona aleatoriamente un subconjunto de las variables del conjunto de datos, dando opciones a variables que normalmente quedarían eclipsadas por otras que tengan mayor relevancia. Al tener un conjunto de datos con muy pocos atributos, tener un número elevado de arboles a preguntar hará que muchas veces se pregunte a arboles repetidos por lo que hay que valorar hasta que punto merece la pena seguir preguntando.

Para realizar este estudio ejecutaremos el modelo con diferentes valores de la variable *n_estimators*[36] (número de arboles en el bosque) y contrastaremos que porcentaje de acierto tiene el modelo con un subconjunto del propio conjunto de datos que conocemos su valor final. Como podemos ver en la figura 6.13a, el valor más optimo del **estimador es 40**, con un **porcentaje de acierto del 61 %**.

Si mostramos la matriz de confusión[33] (figura 6.13b) para poder ver que porcentaje de aciertos tiene el modelo con el conjunto de test, vemos que tiene un **porcentaje de aciertos bastante bajo** llegando a rozar la aleatoriedad.



(a) Calculo del valor n_estimators para el caso de estudio 1.

(b) Matriz de confusión para el modelo Random Forest en el caso de estudio 1.

Figura 6.13: Modelo Random Forest en el caso de estudio 1.

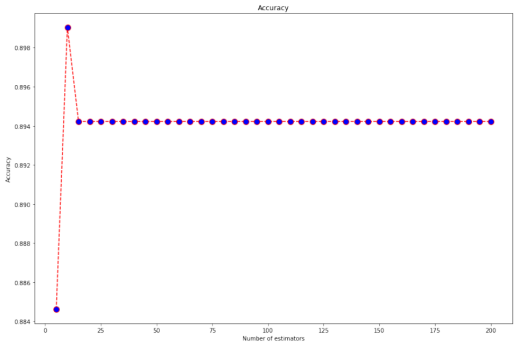
6.4.3. (Caso de estudio 2) Conjunto de datos solo con atributo utilidad definido, añadiendo el mes y año del artículo, eliminando los atributos *gender* y artículo y expandiendo el atributo *respuesta.pubmed_keys*. También escogemos solo los atributos que nos ha indicado como relevantes el caso de estudio 3 del PCA

Igual que en el caso anterior, realizaremos el estudio para detectar a partir de que número de *n_estimators*[36] (número de arboles en el bosque) podemos dar por supuesto que el resultado es de un grupo u otro. A diferencia del caso anterior, podemos ver en la figura 6.14a, el valor más optimo del **estimador es 10**, con un **porcentaje de acierto del 89 %**.

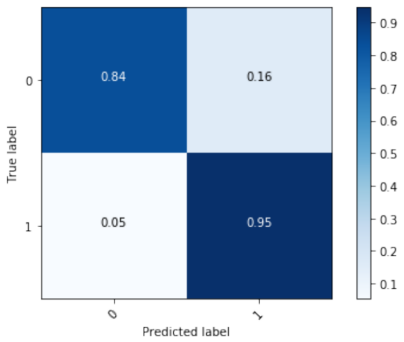
Si mostramos la matriz de confusión[33] (figura 6.14b) para poder ver que porcentaje de aciertos tiene el modelo con el conjunto de test, vemos que tiene un **porcentaje de aciertos bastante aceptable**.

6.4.4. Conclusiones del modelo de Regresión Logística

En este caso, podemos apreciar que el **caso 2 se ajusta a unos resultados aceptables** ya que esta lo suficientemente ajustado para que de resultados razonables sin estar



(a) Calculo del valor n_estimators para el caso de estudio 2.



(b) Matriz de confusión para el modelo Random Forest en el caso de estudio 2.

Figura 6.14: Modelo Random Forest en el caso de estudio 2.

sobreajustado[30]. Por lo que recomendamos al cliente como posible modelo final el modelo de Bosques Aleatorios '*Random Forests*' aplicando las transformaciones al conjunto de datos realizado en el caso de estudio 2.

6.5. Máquinas de vector soporte '*Support Vector Machines*'

6.5.1. Características de los conjuntos de datos a analizar

Al realizar las mismas transformaciones que para el caso de K-NN, las características de los conjuntos de datos son las mismas.

6.5.2. (Caso de estudio 1) Resultados del entrenamiento con el Conjunto de datos solo con atributo utilidad definido y escogiendo los atributos que nos ha indicado como relevantes el caso de estudio 2 del PCA

Para realizar el entrenamiento y posterior validación del modelo de Máquinas de vector soporte '*Support Vector Machines*' es necesario decidir que funciones *kernel* vamos a utilizar para entrenar el modelo. Según la función *kernel* que escojamos, influirá en el comportamiento y resultados del modelo. Para realizar el entrenamiento nos ayudaremos de la librería *svm* del paquete *sklearn* que dispone del modelo de clasificación[37] ya implementado con las funciones *kernel* '*lineal*', '*polynomial*', '*radial (rbf)*' y '*sigmoid*'[38] disponibles. Este paquete también permite poder implementar nuestras propias funciones *kernel* pero descartamos esta opción

por el tiempo que implicaría realizar ese desarrollo, que provocaría que nos saliéramos del tiempo limite para la presentación de este análisis.

Antes de realizar el entrenamiento comparativo, dibujaremos en un gráfico todos los valores siendo 'Y' un atributo en concreto del dataset y 'X' el atributo a predecir. Dibujaremos todos los atributos en el mismo gráfico con la idea de intentar dibujar un hiperplano lineal (división de valores) que nos ayude a poder clasificar los valores del campo utilidad en base los atributos del dataset. Como podemos ver en la figura 6.15 esta división no es posible a simple vista ya que los resultados del campo '*utilidad*' están distribuidos sobre todo el plano de 'X' por lo que descartaremos la utilización de la función kernel *lineal* y probaremos con el resto de funciones.

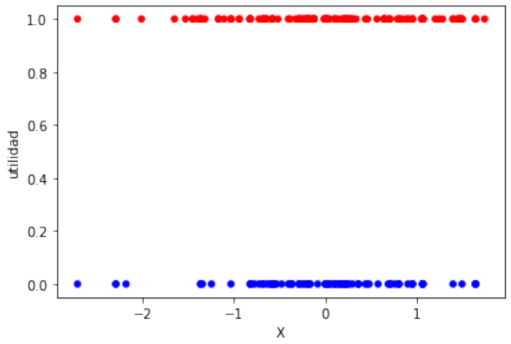
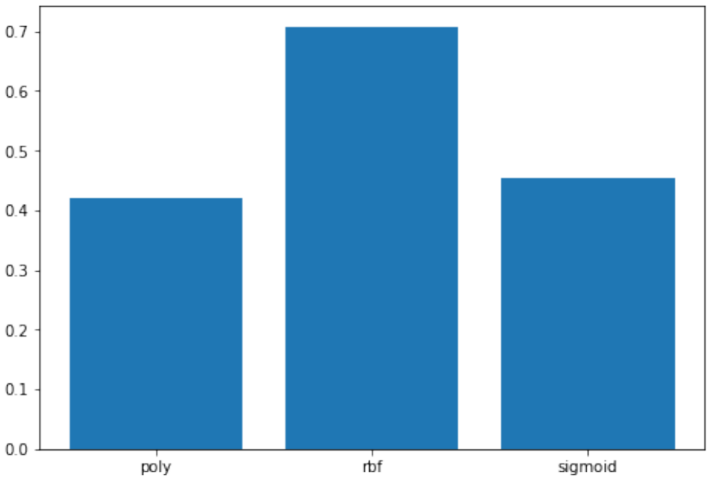


Figura 6.15: Distribución de los valores del dataset respecto al atributo *utilidad* para el caso de estudio 1.

Para explorar los hiper parámetros que necesitaremos, nos ayudaremos de la función '*RandomizedSearchCV*' de la librería '*sklearn.model_selection*' que nos ayudara a ejecutar varias veces el modelo con distintos valores. Para el **parámetro C**, que indica el valor de penalización de los errores en la clasificación (indica el compromiso entre obtener el hiperplano con el margen más grande posible y clasificar el máximo número de ejemplos correctamente). Probaremos **valores aleatorios distribuidos uniformemente entre 1 y 500**. Para el parámetro **gamma** nos basaremos en la propia documentación del modelo que nos indica que utilizando un rango de **valores aleatorios distribuidos uniformemente entre** 10^{-3} **y** 10^{3} suele ser suficiente para encontrar un resultado optimo. Puede consultar el código para la realización de estos entrenamientos en el anexo (Iteración 1).

Una vez realizado el entrenamiento podemos ver (como se aprecia en la figura 6.16) que el modelo con más precisión es el '*radial (rbf)*'. Pero si analizamos su comportamiento con el conjunto de datos de test vemos que su precisión baja obteniendo un **porcentaje de acierto de cerca del 54 %**.

Si mostramos la matriz de confusión[33] (figura 6.17) para poder ver que porcentaje de aciertos tiene el modelo con el conjunto de test, vemos que tiene un **porcentaje de aciertos muy bajo** siendo el resultado totalmente aleatorio.




```
pd.DataFrame(best_score)
```

	kernel	score	params
0	poly	0.419444	{'C': 53.821075955066966, 'gamma': 425.9568020…
1	rbf	0.708333	{'C': 89.593529467132, 'gamma': 484.3701547411…
2	sigmoid	0.452778	{'C': 188.88474765880682, 'gamma': 986.3697466…

Figura 6.16: Resultado del entrenamiento de las diferentes funciones *kernel* para el caso de estudio 1.

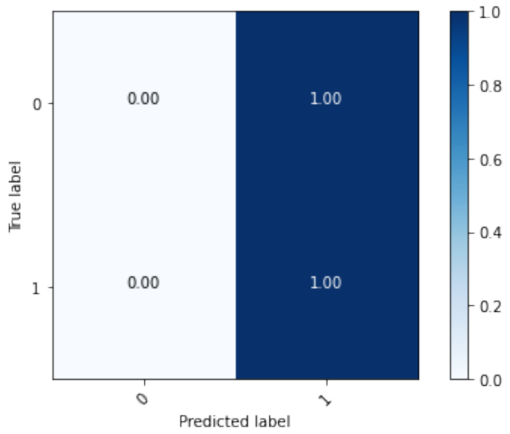


Figura 6.17: Matriz de confusión para el modelo *SVM* con la función *kernel* radial para el caso de estudio 1.

6.5.3. (Caso de estudio 2) Conjunto de datos solo con atributo utilidad definido, añadiendo el mes y año del artículo, eliminando los atributos *gender* y artículo y expandiendo el atributo *respuesta.pubmed_keys*. También escogemos solo los atributos que nos ha indicado como relevantes el caso de estudio 3 del PCA

Igual que en el caso anterior, antes de realizar el entrenamiento del modelo, dibujaremos en un gráfico todos los valores siendo 'Y' un atributo en concreto del dataset y 'X' el atributo a predecir. Igual que en el caso anterior (como podemos ver en la figura 6.18) no es posible hacer la división del hiperplano lineal (división de valores) que nos ayude a poder clasificar los valores del campo utilidad a simple vista, ya que los resultados del campo '*utilidad*' están distribuidos sobre todo el plano de 'X' por lo que descartaremos la utilización de la función *kernel lineal* y probaremos con el resto de funciones.

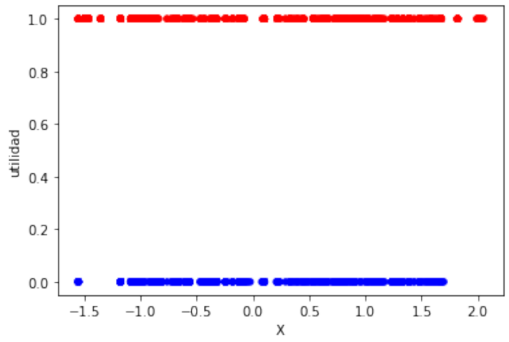
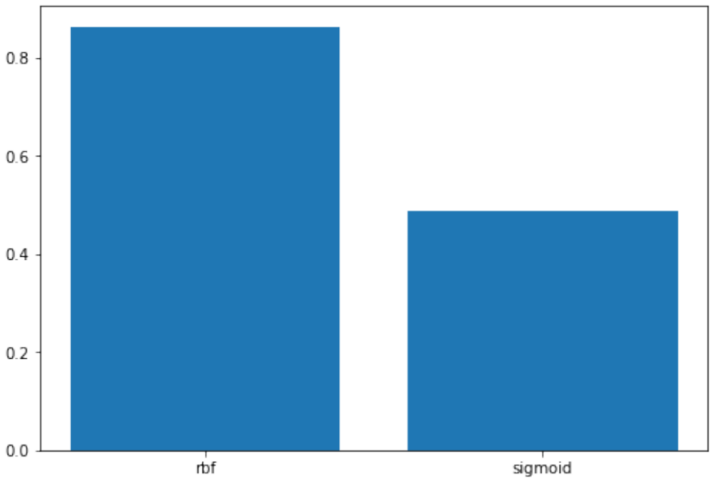


Figura 6.18: Distribución de los valores del dataset respecto al atributo *utilidad* para el caso de estudio 2.

Para explorar los hiper parámetros que utilizaremos los mismos rangos que los utilizados para el caso de estudio 1, para así poder comparar los resultados. **Nota:** Para la función *kernel polinomial* descartamos el poder realizar el entrenamiento por problemas de rendimiento, debido a que el modelo tardaba excesivo tiempo en realizar su entrenamiento (más de 3 días). Puede consultar el código para la realización de estos entrenamientos en el anexo (Iteración 2).

Una vez realizado el entrenamiento podemos ver (como se aprecia en la figura 6.19) que el modelo con más precisión es el '*radial (rbf)*'. Ademas si analizamos su comportamiento con el conjunto de datos de test vemos que su precisión mantiene el **porcentaje de acierto de cerca del 89 %**.

Si mostramos la matriz de confusión[33] (figura 6.20) para poder ver que porcentaje de aciertos tiene el modelo con el conjunto de test, vemos que tiene un **porcentaje de aciertos bastante aceptable**.




```
pd.DataFrame(best_score)
```

	kernel	score	params
0	rbf	0.863327	{'C': 169.2424566596866, 'gamma': 21.685491125…
1	sigmoid	0.488617	{'C': 141.66423772199033, 'gamma': 964.0816236…

Figura 6.19: Resultado del entrenamiento de las diferentes funciones *kernel* para el caso de estudio 2.

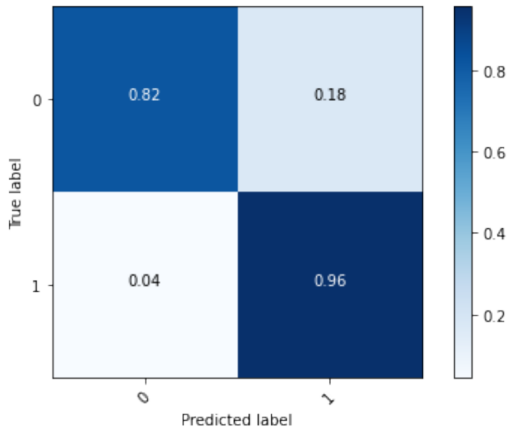


Figura 6.20: Matriz de confusión para el modelo *SVM* con la función *kernel* radial para el caso de estudio 2.

En este caso, podemos apreciar que el **caso 2 se ajusta a unos resultados aceptables** ya que esta lo suficientemente ajustado para que de resultados razonables sin estar

sobreajustado[30]. Por lo que recomendamos al cliente como posible modelo final el modelo de Máquinas de vector soporte '*Support Vector Machines*' con la función *kernel* '*radial (rbf)*' aplicando las transformaciones al conjunto de datos realizado en el caso de estudio 2.

6.6. Comparativa de resultados de los modelos

Una vez realizado el estudio de los tres modelos, podemos ver en la tabla 6.1 que los mejores resultados los obtenemos con las trasformaciones realizadas en el caso de estudio 2 en dos de los 3 modelos (Conjunto de datos solo con atributo utilidad definido, añadiendo el mes y año del artículo, eliminando los atributos *gender* y artículo y expandiendo el atributo *respuesta.pubmed_keys*. También escogemos solo los atributos que nos ha indicado como relevantes el caso de estudio 3 del PCA.).

Ademas el modelo que mejor precisión nos da para ese caso de estudio es el modelo de Bosques aleatorios, seguido de cerca por el modelo de *Support Vector Machine*.

Modelos	Regresión Logística	**Bosques Aleatorios**	Support Vector Machine
Caso de estudio 1	65.78 %	61.53 %	53.85 %
Caso de estudio 2	49.03 %	**89.9 %**	89.42 %

Cuadro 6.1: Comparativa de resultados entre los modelos.

Nota: No añadiremos a la comparativa el modelo K-NN ya que, aunque se podría utilizar como modelo predictivo igual que en los otros casos, en este estudio no se ha entrenado con los mismos atributos que se han entrenado los otros modelos, por lo que no seria justo comparar los resultados con los otros modelos.

Capítulo 7

Conclusiones

7.1. Objetivos Secundarios

En el apartado Análisis de datos **se ha cumplido** el OS1, aplicado las transformaciones necesarias a los datos que tenemos para poder trabajar posteriormente con ellos con los modelos predictivos.

En el apartado Análisis de componentes principales **se ha cumplido** el OS2, pudiendo observar en el apartado de resultados, que hay mucha diferencia según si se usa todas las observaciones o solo las que tienen informado el atributo a predecir (*utilidad*), por lo que se decide junto con el cliente, intentar enriquecer los datos que no tienen informado el atributo (*utilidad*) definiendo este paso como objetivo secundario OS3. Se analizara en el siguiente paso si el resultado de intentar enriquecer estos datos aporta algo al conjunto de datos.

En el apartado Enriquecimiento de los datos **no se ha podido cumplir** el OS3, pudiendo observar en el apartado de resultados que el dataset se encuentra sesgado, por lo que intentar enriquecer los datos lo único que generaría son falsos positivos, perjudicando a los posteriores entrenamientos de modelos predictivos que se estudian en este documento.

En el apartado Regresión Logística **se ha cumplido** el OS4, realizando el estudio de la aplicación del modelo de Regresión Logística, pudiendo observar su bajo nivel de precisión, llegando a desaconsejar su uso con el conjunto de datos actual.

En el apartado Bosques Aleatorios '*Random Forests*' **se ha cumplido** el OS5, realizando el estudio de la aplicación del modelo de Bosques Aleatorios '*Random Forests*', pudiendo observar

un nivel de precisión aceptable para el caso de estudio 2, llegando a aconsejar su uso con el conjunto de datos actual.

En el apartado Máquinas de vector soporte '*Support Vector Machines*' **se ha cumplido** el OS6, realizando el estudio de la aplicación del modelo de Máquinas de vector soporte '*Support Vector Machines*', pudiendo observar un nivel de precisión aceptable para el caso de estudio 2, llegando a aconsejar su uso con el conjunto de datos actual.

En el apartado Comparativa de resultados de los modelos **se ha cumplido** el OS7, mostrando los resultados de los modelos entrenados, pudiendo observar que el modelo con más nivel de precisión es el *Support Vector Machine* para el caso de estudio 2.

Capítulo 8

Anexos

Nota: Puede consultar estos anexos haciendo clic en el enlace o visitando la web de github[39] donde podrá encontrar todos los ficheros del proyecto.

8.1. Procesado de datos

S0 - HOPE extract data.pdf

8.2. Análisis de componentes principales

8.2.1. Iteración 1

S1 - HOPE PCA.pdf

8.2.2. Iteración 2

S1 - HOPE PCA_v2.pdf

8.2.3. Iteración 3

S1 - HOPE PCA_v3.pdf

8.3. Enriquecimiento de los datos

8.3.1. Iteración 1

S2 - HOPE kNN.pdf

8.3.2. Iteración 2

S2 - HOPE kNN_v2.pdf

8.4. Regresión Logística

8.4.1. Iteración 1

S3 - HOPE logistic regression.pdf

8.4.2. Iteración 2

S3 - HOPE logistic regression_V2.pdf

8.5. Bosques Aleatorios 'Random Forest'

8.5.1. Iteración 1

S4 - HOPE random forest.pdf

8.5.2. Iteración 2

S4 - HOPE random forest_V2.pdf

8.6. Máquinas de vector soporte 'Support Vector Machines'.

8.6.1. Iteración 1

S5 - HOPE SVM.pdf

8.6.2. Iteración 2

S5 - HOPE SVM_V2.pdf

Bibliografía

[1] R. J. W. Cline and K. M. Haynes, "Consumer health information seeking on the internet: the state of the art." https://academic.oup.com/her/article/16/6/671/571640, 12 2001.

[2] C. AR, "Los buscadores en la recuperación de información en salud." https://www.medigraphic.com/cgi-bin/new/resumen.cgi?IDARTICULO=35274, 2011.

[3] "Hope project website." http://proyectohope.centromedicodespierta.es/.

[4] C. L. S. Bocanegra, J. L. S. Ramos, C. Rizo, A. Civit, and L. Fernandez-Luque, "Healthrecsys: A semantic content-based recommender system to complement health videos." https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-017-0431-7, 05 2017.

[5] C. Vargas and K. Azucena, "Hope desarrollo de una metodología piloto para vinculación a clinical trials como herramienta de medicina basada en evidencias, para proporcionar medicina centrada en el paciente." http://discovery.uoc.edu/iii/encore/record/C__Rx1033890__Orightresult__U__X2?lang=spi&suite=def, 07 2017.

[6] "Ruido en los datos (noisy data)." https://en.wikipedia.org/wiki/Noisy_data.

[7] "Documentación api pubmed." https://www.ncbi.nlm.nih.gov/CBBresearch/Lu/Demo/tmTools/RESTfulAPIs.html.

[8] "Definición de pca (analisis de componentes principales." https://es.wikipedia.org/wiki/An%C3%A1lisis_de_componentes_principales.

[9] "Definición de k-nn (k vecinos más próximos)." https://es.wikipedia.org/wiki/K_vecinos_m%C3%A1s_pr%C3%B3ximos.

[10] "Why is scaling required in knn and k-means?." https://medium.com/analytics-vidhya/why-is-scaling-required-in-knn-and-k-means-8129e4d88ed7.

[11] "Definición de regresión logística." https://es.wikipedia.org/wiki/Regresi%C3%B3n_log%C3%ADstica.

[12] S. Sperandei, "Understanding logistic regression analysis." https://hrcak.srce.hr/115732, 2014.

[13] "Definición de random forest." https://es.wikipedia.org/wiki/Random_forest.

[14] "Definición de máquinas de vectores de soporte." https://es.wikipedia.org/wiki/M%C3%A1quinas_de_vectores_de_soporte.

[15] "Definición de función kernel." https://en.wikipedia.org/wiki/Kernel_method.

[16] "Publicación de kernels para utilizar en máquinas de vectores de soporte." http://www.kernel-machines.org/publications.

[17] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions." https://ieeexplore.ieee.org/abstract/document/1423975, 04 2005.

[18] S.-J. Yoon and J.-H. Kim, "Is the internet more effective than traditional media? factors affecting the choice of media." http://www.journalofadvertisingresearch.com/content/41/6/53, 11 2001.

[19] C. MaféS and S. Blas, "Implicaciones del uso de buscadores en el comportamiento de compra online." https://www.sciencedirect.com/science/article/pii/S1135252312601014, 02 2009.

[20] V. M. and T. K., *History and Overview of the Recommender Systems*. Vishal Bhatnagar, IGI Global, 2017.

[21] A. aron van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation." http://papers.nips.cc/paper/5004-deep-content-based-music-recommendation.pdf, 2013.

[22] D. Chong, "Deep dive into netflix's recommender system." https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48, 04 2020.

[23] S. Bhoi, L. M. Li, and W. Hsu, "Premier: Personalized recommendation for medical prescriptions from electronic records." https://arxiv.org/abs/2008.13569, 08 2020.

[24] J. C. Clauser, J. Maas, J. Arens, T. Schmitz-Rode, U. Steinseifer, and B. Berkels, "Automation of hemocompatibility analysis using image segmentation and a random forest." https://arxiv.org/abs/2010.06245, 10 2020.

[25] "Pubmed website." https://pubmed.ncbi.nlm.nih.gov/advanced/.

[26] "Clinicaltrials website." https://clinicaltrials.gov.

[27] "Medlineplus website." https://medlineplus.gov.

[28] T. Yamagata, A. O'Kane, A. Ayobi, D. Katz, K. Stawarz, P. Marshall, P. Flach, and R. Santos-Rodríguez, "Model-based reinforcement learning for type 1diabetes blood glucose control." https://arxiv.org/abs/2010.06266, 10 2020.

[29] C. S. Figueroa, P. C. Vázquez, I. T. Martín, and S. del Rey, "El sesgo de selección muestral." http://www.ahepe.es/VICongreso/descargas/Cristina_Sanchez_Figueroa.pdf, 2012.

[30] "Que es el sobreajuste (en inglés overfitting)." https://es.wikipedia.org/wiki/Sobreajuste.

[31] "sklearn.model_selection.train_test_split." https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html.

[32] "Documentación del algoritmo kneighborsclassifier." https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html.

[33] "Confusion matrix." http://www2.cs.uregina.ca/~dbd/cs831/notes/confusion_matrix/confusion_matrix.html.

[34] "Que es la validación cruzada." https://es.wikipedia.org/wiki/Validaci%C3%B3n_cruzada.

[35] "Bootstrap aggregating (bagging)." https://en.wikipedia.org/wiki/Bootstrap_aggregating.

[36] "sklearn.ensemble.randomforestclassifier." https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

[37] "sklearn.svm.svc." https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.

[38] "Funciones kernel disponibles en el paquete sklearn." https://scikit-learn.org/stable/modules/svm.html#svm-kernels.

[39] "Repositorio del código de proyecto." https://github.com/neburs/HOPE_rank_predictor.