*Optimización de variables en modelos predictivos*

Eduardo Miguel Botía Domingo   
*dpto. Ciencias de la Computación e Inteligencia Artificial*  
*Universidad de Sevilla*Sevilla, España  
[edubotdom@alum.us.es](mailto:edubotdom@alum.us.es)

Isaac Muñiz Valverde  
*dpto. Ciencias de la Computación e Inteligencia Artificial*  
*Universidad de Sevilla*Sevilla, España  
[isamunval@alum.us.es](mailto:isamunval@alum.us.es)

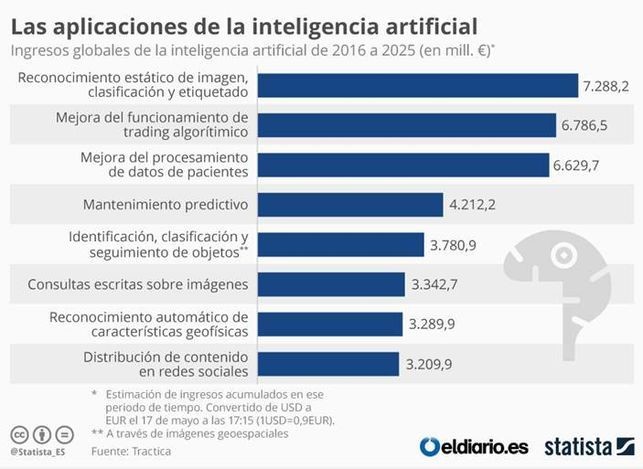
*Resumen*— El objetivo fundamental de este trabajo busca realizar una implementación que permita obtener de forma eficaz y que pueda ser medida de forma objetiva, mediante métodos de envoltura utilizando varias estrategias de búsqueda secuencial, un subconjunto de variables de un conjunto de datos original sobre las que procesar un algoritmo de aprendizaje automático, de tal forma que podamos optimizar su rendimiento y coste computacional sin penalizar en exceso su tasa de predicción o acierto en el proceso. Además, busca estudiar su impacto en su efectividad y tasa de predicción, además de implementar un algoritmo que permita evaluar de forma mesurable, robusta e independiente al contexto el impacto sobre la tasa de acierto que tiene realizar cada modificación en el conjunto.

Tras la elaboración del trabajo, hemos obtenido como respuesta una implementación eficiente y que ofrece un alto rendimiento utilizando técnicas de trabajo en paralelo que hacen eficiente una serie de métodos que realizan dicha optimización, mostrando por pantalla de manera explícita y gráfica la relación entre los mejores conjuntos de variables según su tamaño y su tasa de acierto utilizando un mecanismo robusto de validación que ha sido un producto adicional obtenido tras el trabajo sobre la materia principal del trabajo. Como conclusión adicional, hemos obtenido que no siempre los conjuntos más amplios de variables no son los que ofrecen una mayor tasa de acierto por fenómenos como el sobreajuste o por variables que introducen ruido o no tienen tanta representatividad en la variable sobre la que el algoritmo toma la decisión, lo cual añadido a la complejidad mayor que conlleva procesarlos; nos reafirma la importancia de establecer una optimización previa sobre el conjunto de variables que permita reducir el coste de computación y la eficacia del algoritmo.

Palabras Clave—Inteligencia Artificial, aprendizaje automático, métodos de envoltura, sobreajuste, aprendizaje supervisado, variable, selección de características, optimización, coste computacional, árboles de decisión, validación cruzada, paralelización, búsqueda secuencial, conjunto de características, ruido.

# Introducción

El campo de la Inteligencia Artificial es un área de la informática que, sin ser relativamente reciente, cada vez cuenta con un mayor auge en el contexto actual, y se prevé que se convierta en un motor económico a nivel mundial en los próximos años y que su avance pueda introducir cambios sustanciales en la forma de vida de la población, así como en casi cualquier disciplina como puedan ser la publicidad, comercio, medicina, transporte o investigación, mediante tareas como el tratamiento de datos y su procesamiento, así como múltiples aplicaciones de todo tipo. Esta revolución por llegar exige que se realice un trabajo exhaustivo y que exista un campo de investigación sustancial.



1. Aplicaciones y previsión de ingresos en campos de inteligencia artificial del año 2016 a 2025. Imágen con derechos Creative Commons, por Tractica y distribuida por Statista.

En este contexto, una de las mayores áreas de trabajo se encuentra en el aprendizaje automático, que se basa en construir software que mejore de forma autónoma mediante el procesamiento de información que se le proporcione, y permite construir bases de conocimiento a partir de dichos datos, clasificación y diagnóstico, minería de datos, o establecer patrones en datos hasta el momento ocultas y resolver problemas mediante planificación.

Una de las ramas o que destacan dentro de esta disciplina, se trata del aprendizaje supervisado, el cual, facilitándole una serie de casos, datos o información supervisada y correcta, compuesta por características, variables o circunstancias en las que se produjeron nos permita encontrar patrones y nos permita estimar o predecir el resultado u otra variable que denominamos objetivo, la cual es el objetivo de esta estrategia. Dentro de esta rama, existen múltiples técnicas y algoritmos específicos para implementar la teoría que hemos expuesto, como el aprendizaje mediante árboles de decisión, los cuales serán una parte indispensable de nuestra implementación por defecto salvo que el usuario prefiera especificar otro algoritmo de aprendizaje.

Entrando en materia de implementación, para el trabajo con este tipo de técnicas, cuando se trabaja con conjuntos pequeños de casos o pocas variables, no toma gran relevancia realizar una optimización de recursos, ya que no suponen un problema, aunque, en campos de trabajo y escenarios en los que se utilizan estas técnicas reales y en la práctica, cuando se trata de procesar y trabajar con una enorme cantidad de datos debe tenerse cuidado, no puede considerarse como un campo de trabajo trivial. Es por ello que existen ciertas técnicas y campos de estudio que buscan realizar una optimización de recursos mediante la selección de las variables más representativas y de mayor relevancia con respecto al resultado final, o la decisión, que componen el conjunto de información con la que se trabaja, lo cual no empeora necesariamente la tasa de acierto del algoritmo, lo cual puede resultar ser beneficioso en todos los sentidos, permitiendo obtener resultados más fiables a un menor coste, a expensas de realizar un análisis previo del conjunto de datos.

Como especificamos en el resumen anteriormente presentado, surgiendo esta necesidad, el grupo de trabajo tomó como decisión elegir el trabajo propuesto por el equipo docente de la asignatura Inteligencia Artificial, que toma como objetivo la selección de características o variables predictoras del conjunto de variables mediante métodos de envoltura, que nos permitan obtener un resultado más preciso. Para ello, pretendemos desarrollar funciones basadas en métodos de búsqueda secuencial. Además del objetivo nominal del trabajo, buscaremos obtener una mayor tasa de acierto, una mayor sencillez del modelo predictivo, lo cual permite interpretar de manera más intuitiva los resultados y predicciones que realizará el algoritmo; junto a una reducción generalizada del coste de computación eliminando las variables que introduzcan ruido y distorsionen el resultado final; además de desarrollar una técnica de evaluación de soluciones robusta que nos permita valorar objetivamente el grado de acierto sobre un subconjunto de variables del original.

En este documento se pretende documentar con todo detalle el enfoque de este trabajo, información preliminar acerca de los métodos empleados, la metodología, la implementación realizada, análisis de pruebas, experimentos, resultados, métricas y observaciones que se han llevado a cabo, además de las fuentes de información y conocimiento que han basado la realización del presente trabajo. Por otra parte, se ofrece en un fichero notebook, escrito utilizando el software Jupyter notebook y en lenguaje Python, las funciones que componen una implementación como respuesta al problema planteado, junto con pruebas, explicaciones teóricas acerca del problema, e información acerca de cada decisión tomada a bajo nivel y explicando el propio código, e interpretaciones acerca de los resultados obtenidos.

# Preliminares

Para poner en contexto de los métodos y técnicas que se pretenden llevar a cabo en este trabajo y poder fundamentar la implementación de estos, en esta sección buscamos realizar una introducción de las técnicas empleadas y también trabajos relacionados, si los hay.

## Métodos empleados

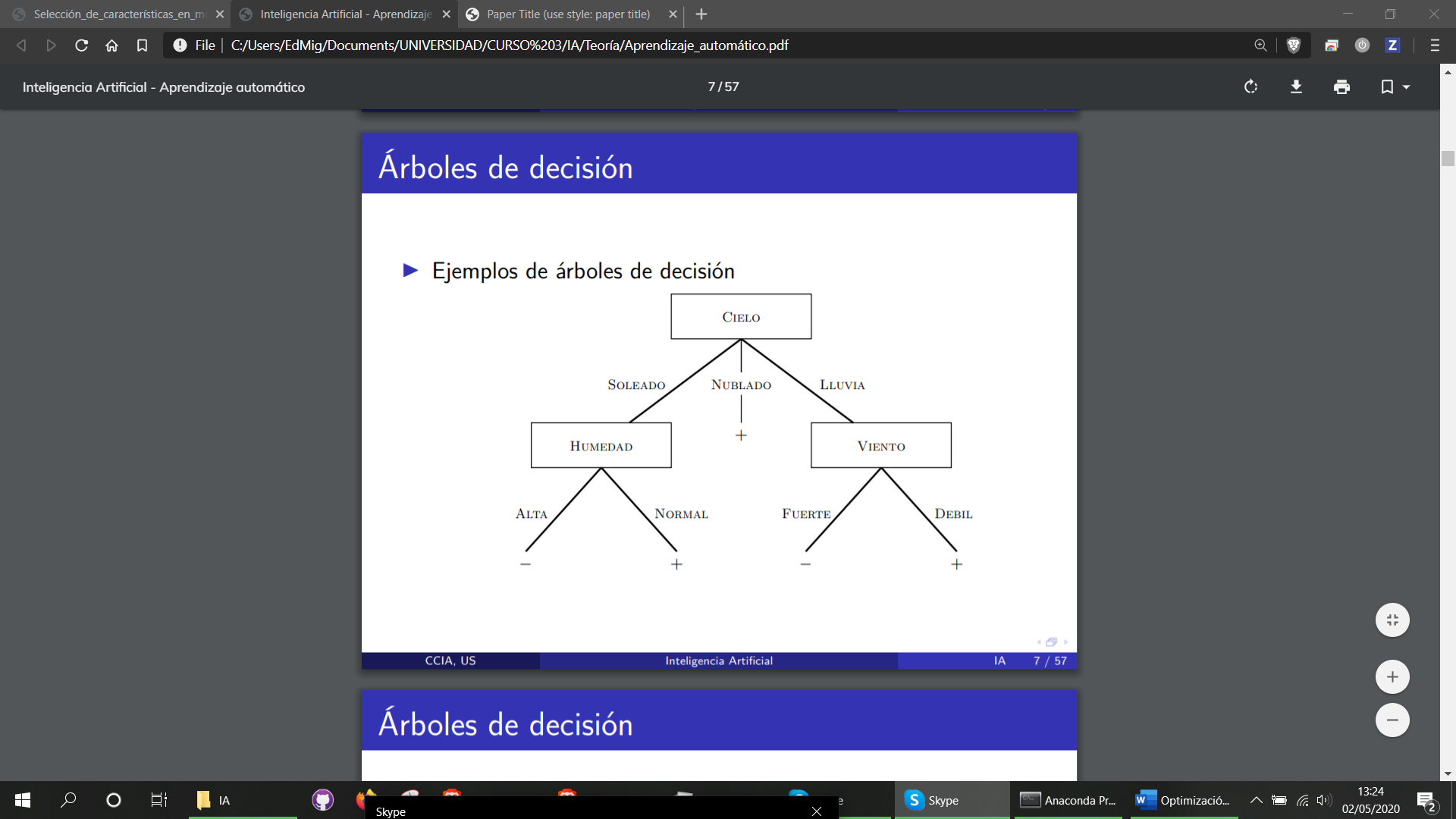
* Algoritmos de árboles de decisión

Se trata de una técnica de aprendizaje supervisado que nos permite intuitiva y visualmente implementar este tipo de aprendizaje, y que además nos ofrece obtener una función objetivo relativamente de manera sencilla.

Se obtiene, como el resto de las técnicas de aprendizaje supervisado, a partir de un conjunto de ejemplos de entrenamiento, de tal manera que el árbol “crece” en anchura y profundidad. Como podemos apreciar en la posterior ilustración, se encuentran compuestos por nodos interiores, que son los atributos o variables del conjunto de datos; un número finito de arcos, que representan los valores que puede tomar un nodo y hojas, que son valores de clasificación, binarios o no, que contendrán el resultado de la predicción.

Además, de una forma intuitiva, nos permiten conocer en forma de lógica proposicional, una función objetivo que nos permitirá obtener una predicción.

A continuación, mostramos un ejemplo de árbol en el que se pueden apreciar todos estos componentes.



1. Representación de un árbol de decisión. Apuntes de aprendizaje automático por la asignatura Inteligencia Artificial (IS).

Existe más de un algoritmo para implementar estos árboles, como ID3, que ofrece una implementación más sencilla cuando se trabaja a mano, aunque en concreto, nos centraremos el algoritmo CART, que además de trabajar con variables categóricas como permite ID3, nos permite utilizar variables continuas.

De acuerdo con la exposición realizada por el Doctor Jorge Martín Arevaillo, las principales características de este método, es su *origen estadístico*, que trabaja con variables de todo tipo sin *discretizar las variables continuas*, que el corte de cada nodo es binario y que da lugar a estructuras de árbol de mayor profundidad, con un criterio de corte que conduce al *mayor decrecimiento de la impureza* y con uno de parada que propone *segmentar la base de datos hasta obtener una estructura de árbol lo más compleja posible*, declarando un *nodo como terminal cuando su tamaño es inferior a un umbral* muy pequeño. Su *complejidad se mide por el número de nodos terminales* y *se poda la estructura del árbol maximal obtenido*.

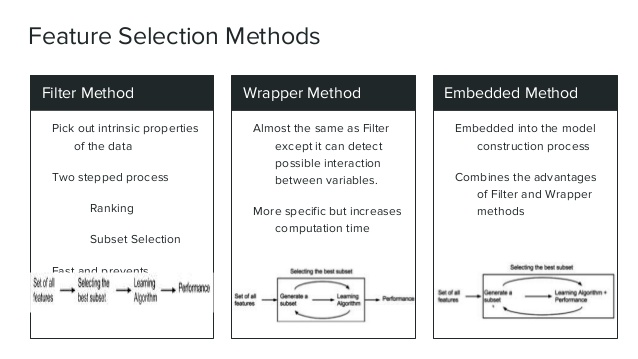
* Métodos de envoltura

Conviene conocer previo a la consulta del presente trabajo, que, existe una clasificación generalizada por la gran cantidad de técnicas y paradigmas de selección de características que existen, como son los métodos de filtro, en los cuales las variables y resultados son analizados mediante test estadísticos; los métodos integrados, que son aquellos que tienen su propia implementación de selección de las variables que más aportan información a la predicción, decidiendo si la variable es mejor o peor.

Finalmente, existen los métodos de envoltura, que son aquellos en los que el algoritmo de entrenamiento tiene que ver con el proceso de entrenamiento de las características, es decir, no se realiza un estudio estadístico de las variables independiente, ni tampoco se integra en el mismo proceso. Para ello definimos subconjuntos de variables del conjunto original y evaluamos los resultados tras aplicar un proceso de entrenamiento con las variables seleccionadas, para después aplicar un procedimiento de validación y mediante casos de prueba comprobar la eficacia del algoritmo. En nuestro caso, se ofrece al usuario elegir el método de aprendizaje que estime oportuno, aunque por defecto, el algoritmo que acompañará al proceso anteriormente descrito será el de creación de árboles.

De manera adicional, debemos apuntar a la posibilidad de que se combinen más de uno de los métodos anteriores para desarrollar uno que obtenga un mejor compendio entre eficiencia, rendimiento y precisión.

En la siguiente imagen podemos apreciar los diferentes tipos de métodos de selección y un pequeño esquema que aclara de manera intuitiva su funcionamiento.



1. Clasificación de los métodos de selección de características. Imagen de Bhargav Srinivasan de la presentación “Feature Selection of Medical Diagnosis Data”.

* Evaluación de soluciones robusta: Validación cruzada

Para llevar a cabo la implementación de este problema, requerimos como parte fundamental el realizar una estimación de la capacidad predictiva de un algoritmo de aprendizaje, dadas cada una de las soluciones posibles, o combinaciones de variables en el subconjunto del total que componen los casos de entrenamiento y pruebas.

Para llevar a cabo esta evaluación o predicción, no obstante, debemos tomar en cuenta que los algoritmos de aprendizaje automático tienen un componente de aleatoriedad en sus resultados, por lo que, sin ser incorrectos, pueden arrojar diferentes resultados tras su ejecución y por ende, tras su evaluación en múltiples ocasiones. Por otra parte, dependiendo del propio conjunto de pruebas y de entrenamiento, o qué casos son definidos para cada una de las dos fases, el resultado de aplicar el algoritmo puede verse modificado.

Estas circunstancias, lejos de ser una trivialidad, dificultan la tarea de realizar una estimación, y es por ello, que recurriremos al método de validación cruzada, frente a otros más básicos y con menor coste computacional como un simple particionado, aunque a costa de una precisión mucho menor en los resultados. Realizar este método nos permitirá reducir la aleatoriedad de los resultados y que puedan verse tan afectados causados por la elección de uno u otro conjunto de casos de prueba y de entrenamiento, como hemos comentado anteriormente.

En primer lugar, este método debe especificársele el número de pliegues deseados. Esto es, que dividirá un conjunto unificado de datos entre el número de pliegues especificado. Tras esto, realizará ese mismo número de evaluaciones, de tal manera que en cada de ellas sea un subconjunto de datos que hemos dividido antes el conjunto de prueba, y el resto de entrenamiento. Este método de validación cruzada no consiste en más que realizar estas iteraciones para luego realizar una media de los resultados obtenidos y así obtener un resultado.

A continuación, mostramos una imagen intuitiva que muestra las iteraciones que realiza el algoritmo, y su posterior tratamiento de los resultados:



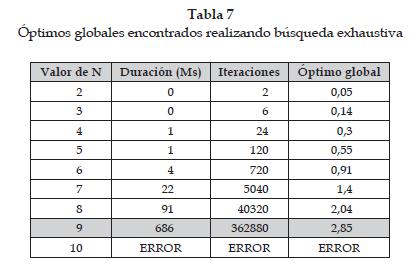
1. “A TensorFlow example using the Keras API” por Brandon M. Greenwell and Bradley C. Boehmke.

Finalmente, para evitar el factor aleatoriedad, tomamos una solución similar a la anterior, y es la estimación de resultados. Para ello, llevamos a cabo sucesivas iteraciones de esta validación cruzada, para finalmente, realizar una media con todos los resultados, y así obtener un resultado más preciso. Este parámetro será elegido por el usuario al utilizar esta función, ya que aumentará de manera más que apreciable la complejidad y coste computacional del algoritmo.

* Algoritmo y estrategias de búsqueda secuencial para selección de características

Anteriormente en este documento, hemos mostrado la importancia de aplicar estos métodos y una breve introducción a los métodos, aunque cabe hacer un breve análisis de estos y sus opciones, concretamente las que se proponen realizar como respuesta a la solicitud de implementación del algoritmo.

Debemos descartar en cualquier caso la búsqueda exhaustiva, o componer todos los conjuntos posibles de variables que pudieren existir y evaluarlos todos para conseguir el mejor, pero es inviable en casos con gran cantidad de combinaciones. En la siguiente imagen mostramos el resultado de aplicar este método de búsqueda, en detrimento del que proponemos, según el número de variables.



1. Óptimos globales encontrados realizando una búsqueda exhaustiva. Por Elías D. Niño, Carlos J. Ardila, del artículo “Algorithm based on finite automata for obtaining global optimum combinatorial problems” de la obra online Ingeniería y Desarrollo con ISSN 2145-9371.

Haciendo una breve síntesis, estos métodos implementan una búsqueda de las variables más importantes y representativas en función del tamaño del conjunto y en especial, por su eficacia o tasa de acierto, la cual especificaremos más tarde.

Para ello y para cada posibilidad que se plantee como válida, realizan un proceso de entrenamiento sobre un conjunto de variables concreto, para que después, mediante un método de validación robusta como el que hemos mostrado anteriormente, puedan evaluar la eficacia de cada conjunto, quedándose con el mejor por cada uno de los tamaños del subconjunto y anotando su puntuación, para después poder consensuar cuál es el mejor conjunto atendiendo también factores como la complejidad que provoca un elevado número de variables en relación con una tasa de acierto similar para un conjunto menor, y por ende, menos complejo.

Para proponer estos conjuntos de variables deberá realizarse previamente una búsqueda de dichas variables, de tal manera que puedan ser evaluadas posteriormente y elegir las mejores. Por ello, existen diferentes maneras de realizarlas, desde métodos que implementan metaheurística o algoritmos genéricos, que son más elaborados y con mayor eficiencia, u otras más simples como la que usaremos, las secuenciales. Estas búsquedas, de manera genérica, también llamadas como lineales, buscan localizar un elemento de una lista de manera lineal como su nombre indica recorriendo paso a paso las posibilidades disponibles, siguiendo el orden de elementos de la propia lista, mientras que otros métodos ofrecen mayor eficacia y eficiencia cuando la lista sigue algún orden y sus valores por los que se busca se encuentran ordenados [8]

Existen diferentes enfoques teóricos y prácticos para llevarlas a cabo, como son hacia adelante, que son aquellas que parten de un conjunto vacío y proceden a ir rellenándolo con las variables con respecto las van “encontrando” o seleccionando la mejor opción disponible, así como las del enfoque hacia atrás que resulta ser todo lo contrario, van reduciendo las variables que existen en el subconjunto, eliminando una a una las peores que haya. Por último, se encuentran las mixtas, que no hacen sino combinar ambos enfoques, de tal manera que busca añadir una variable al conjunto para después buscar la peor para eliminarla del conjunto. Como dato, podemos apreciar que estas estrategias no son únicas en este campo, ya que en otros como en estadística [6] y [7], también pueden aparecer para seleccionar variables para un modelo de regresión lineal múltiple.

En nuestra implementación, hemos elegido por simplicidad implementar una función basado en la estrategia secuencial hacia adelante y, posteriormente, añadir la opción mixta, porque consideramos que puede ser la que ofrezca a priori resultados más exactos o interesantes para las pruebas sobre conjuntos.

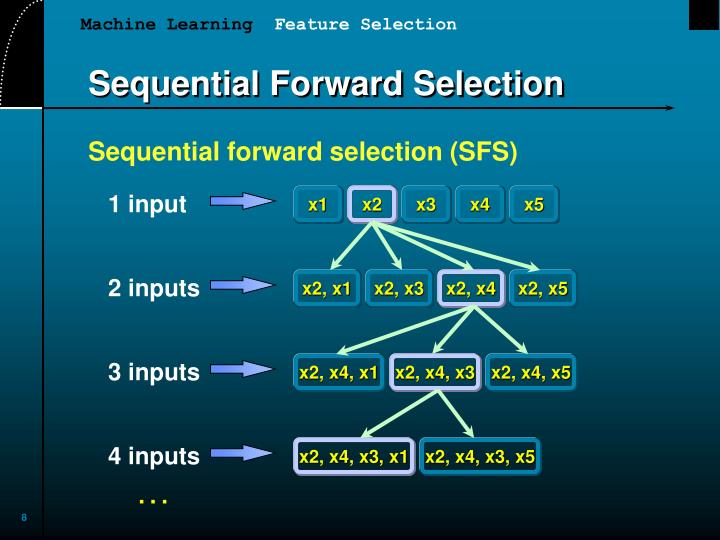
Analicemos los dos algoritmos seleccionados para ser desarrollados en la implementación del algoritmo.

* 1. Sequential forward selection (SFS)

Es un algoritmo que trabaja mediante el paradigma de la búsqueda secuencial hacia adelante por definición. Se inicializa un conjunto de variables vacío y en cada iteración selecciona entre las variables pendientes que no hayan sido tratadas todavía, la mejor que esté disponible. El algoritmo tiene como condición de parada que se hayan añadido todas las variables del conjunto, o el número deseado, que será requerido como parámetro; y devolverá una tabla por cada una de las soluciones que encuentre el método, una por tamaño.

Para cada una de esas adiciones a la lista, se realizará la evaluación de todas las variables que puedan ser añadidas, y se elegirá la mejor posibilidad acorde a la valoración que le haya otorgado el método de evaluación.

Más información e implementación en el software MATLAB [9]. Se adjunta una imagen aclarativa adicional:



1. . “Feature Selection for Pattern Recognition” por J.-S. Roger Jang, de la National Taiwan University.
   1. Sequential floating forward selection (SFFS)

Si bien el anterior algoritmo se trataba de una búsqueda secuencial hacia adelante, este se trata de una búsqueda secuencial mixta, que toma su estrategia la base de la búsqueda hacia adelante, inicializando la búsqueda desde una lista vacía, para la cual se irán añadiendo las mejores opciones de forma lineal, tras evaluar las mejores una a una, con la modificación que implica que, tras la adición de una variable nueva al conjunto de solución, el algoritmo buscará si es posible eliminar alguna variable, la peor, de tal manera que el conjunto ofrezca mejor rendimiento cuando es eliminada que cuando estaba en el conjunto.

El algoritmo tendrá como condición de parada que la solución se encuentre estabilizada, es decir, que todas las variables hayan sido añadidas una vez y que no puedan ser eliminadas más. Esta última variable es interpretable, por lo que la dejaremos a elección del usuario al invocar la función.

## Trabajo Relacionado

Consideramos interesantes y relevantes trabajos realizados sobre la materia de la cual hemos extraído información y creemos que pueden aportar más información a aquellos que deseen más información o puntos de vista acerca de esta temática.

* “Feature Selection of Medical Diagnosis Data Using Genetic Algorithm and Data Mining” - <https://www.slideshare.net/buggytarantino/medical-data-diagnosis-59467371>
* “Feature selection using Wrapper methods in Python” <https://towardsdatascience.com/feature-selection-using-wrapper-methods-in-python-f0d352b346f>
* “Feature Selection for Pattern Recognition” por J.-S. Roger Jang, de la National Taiwan University”. <https://www.slideserve.com/burian/feature-selection-for-pattern-recognition>

# Metodología

En esta sección, pretendemos realizar una descripción de los métodos implementados, mostrándolos y explicándolos en profundidad, tanto la funcionalidad básica, así como los añadidos adicionales sugeridos en la propuesta del trabajo. Cabe mencionar, como hicimos en la introducción a este documento, que este proyecto, cuenta con un fichero notebook con una introducción, la implementación y una explicación pormenorizada de cada elemento que componen las funciones implementadas, sus dependencias, decisiones de diseño e importaciones que son necesarias para el correcto funcionamiento del método; así como las pruebas con un breve análisis de circunstancias que pueden apreciarse tras las pruebas.

A continuación, analizamos uno a uno cada método implementados.

### Implementación del método de evaluación.

Como parte fundamental y utilizada por los procedimientos objetivo de este trabajo, SFS, de búsqueda secuencial hacia adelante y SFFS, de búsqueda mixta hacia adelante como estrategia primaria y hacia atrás; en primer lugar, debe diseñarse la implementación de un método de evaluación robusta, el cual se basa en una validación cruzada, cuyo fundamento se encuentra explicado más detalladamente en la sección anterior en su correspondiente apartado de este documento.

Básicamente, buscamos implementar un algoritmo que realice una evaluación basada fundamentalmente en la tasa de acierto y error producidos por un algoritmo de aprendizaje automático utilizando un conjunto de variables reducido con respecto al original. Gracias a él, podremos saber si un conjunto de variables es mejor o peor que otro, lo cual es necesario para la implementación de dichos algoritmos.

En lo que respecta a la implementación, pasamos a explicarla paso por paso mediante comentarios en la implementación del algoritmo.

Con el objetivo de optimizar el funcionamiento y rendimiento de este algoritmo, y aún más importante, de todos los que basan buena parte de su funcionamiento y lo utilizan posteriormente, tras el diseño inicial del método se procedió a modificar la anterior función para realizar paralelización en el bucle que realiza la validación cruzada.

A continuación, mostramos la implementación del método.

**Procedimiento evaluar\_solucion\_paralelizada**

**Entrada**:

* datos : conjunto de datos procesados de origen externo utilizando la librería Pandas.
* variablesEscogidas : conjunto de variables del conjunto de datos original que pretendemos evaluar, para averiguar su eficacia al aplicar un algoritmo de aprendizaje sobre él.
* N\_exp : número de veces que repetiremos el experimento, para obtener una evaluación más adecuada y equilibrada, a costa de una mayor complejidad.
* CV : número de pliegues que realizará el algoritmo de validación cruzada sobre el conjunto de datos original. Se realizará ese número de veces de estimaciones, dividiendo el conjunto de pruebas por ese mismo número, siendo uno de prueba y el resto de entrenamiento.
* estimador: de manera adicional a la propuesta del trabajo, se añade la posibilidad algoritmo de aprendizaje automático deseado por el usuario. Por defecto, se utilizará el árbol de decisión binario.
* métrica\_evaluación: nos permitirá modificar la métrica de evaluación a usar. Es opcional, ya que por defecto utiliza 'balanced\_accuracy'

**Salida**:

* Tasa de aciertos balanceada promedio obtenida.

**Algoritmo**:

1. Seleccionar del conjunto de datos de entrada, el subconjunto de columnas (variables) que queremos evaluar.
2. Repetir N\_Exp veces y promediar el resultado
   1. Realizar experimento de valización cruzada (siendo CV el número de folds) mediante la función ‘cross\_val\_score’.
3. Devolver el resultado promedio.

### Implementación de la Sequential Forward Selection:

Con la idea de conseguir un conjunto de las mejores variables, que predigan mejor el resultado, utilizaremos un algoritmo de búsqueda, conocido como Sequential Forward Selection, o SFS, que obtiene buenos resultados sin tener que recurrir a métodos de complejidad exponencial, como la búsqueda exhaustiva.

Comenzando por un conjunto de variables, trata de escoger la mejor variable diferente a las que ya contiene, para ello, evalúa todas aquellas candidatas mediante el método de evaluación robusto que hemos presentado anteriormente. Partiendo de un conjunto vacío, finalmente obtendrá como salida una tabla con las mejores variables, su puntuación, y su tamaño, que es definido por el número de sus variables.

Para ello, debemos pasarle el conjunto de datos con las propiedades que desean probarse y la variable objetivo en última posición, seleccionados sobre el conjunto de datos original. Además, deberá facilitarse el número de variables que desean ser probadas del conjunto de datos, por si se desea probar un subconjunto menor. Por defecto, este parámetro será igual a la longitud de propiedades del conjunto de datos, con el fin de que realice el proceso con todas las variables. A su vez, también se le pueden pasar de forma opcional los parámetros que nombramos anteriormente, para decidir el algoritmo de aprendizaje y la métrica de evaluación deseada. Procedemos a mostrar un esquema del procedimiento.

**Procedimiento sequential\_forward\_selection**

**Entrada**:

* datos\_seleccionados : conjunto de datos preprocesados con las variables, seleccionadas previamente, que desean estudiarse.
* D :.Por defecto toma valor 0, y es el número de variables del conjunto anterior que se desean procesar.
* estimador: de manera adicional a la propuesta del trabajo, se añade la posibilidad algoritmo de aprendizaje automático deseado por el usuario. Por defecto, se utilizará el árbol de decisión binario.
* métrica\_evaluación: nos permitirá modificar la métrica de evaluación a usar. Es opcional, ya que por defecto utiliza 'balanced\_accuracy'

**Salida**:

* Tabla con cada una de las combinaciones obtenidas en cada iteración, su tamaño y su rendimiento.

**Inicialización**

* SolucionActual: Almacena el mejor conjunto de variables obtenido en cada iteración. Inicialmente está vacío.
* K=0. K es el contador de iteraciones o de variables seleccionadas en cada iteración.

**Algoritmo**:

1. Mientras que K<D:
   1. Seleccionar y añadir la mejor variable V del conjunto original de variables que no se encuentre en SolucionActual.
      1. SolucionTemporal = SolucionActual + V
      2. Evaluar SolucionTemporal y guardar su rendimiento.
   2. Seleccionar la mejor SoluciónTemporal y hacer SolucionActual = MejorSolucionTemporal y K = K+1
2. Devolver tabla con cada una de las MejorSolucionTemporal, el tamaño y el rendimiento de cada una.

### Implementación de la Sequential Floating Forward Selection:

Como alternativa al método anterior, utilizaremos un algoritmo de búsqueda más avanzado, secuencial mixto que combina la estrategia hacia adelante que utilizamos con el algoritmo SFS, y atrás como secundaria.

Su implementación es muy similar al SFS, con la adición de una evaluación que nos permitirá conocer si resulta más óptimo realizar la adición o eliminación. Para compararlo, evaluaremos ambas posibilidades utilizando el método de evaluación robusto que presentamos anteriormente. Como diferencia, este algoritmo puede tener la circunstancia de que una vez se han añadido todas las variables, quizá resulta óptima la solución de eliminar una variable, por lo que no puede acabar cuando se han recorrido todas las variables. En lugar de ello, debemos establecer una condición de parada que consista en que, efectivamente, todas las variables hayan sido tratadas por el algoritmo, y que hayan pasado un número concreto de evaluaciones adicionales para que el algoritmo "tenga tiempo" de eliminar las variables que resulten prescindibles. Procedemos a mostrar un esquema del procedimiento.

**Procedimiento sequential\_floating\_forward\_selection**

**Entrada**:

* datos\_seleccionados : Ídem algoritmo anterior.
* estimador: Ídem algoritmo anterior.
* métrica\_evaluación: Ídem algoritmo anterior.

**Salida**:

* Tabla con cada una de las combinaciones obtenidas en cada iteración, su tamaño y su rendimiento.

**Inicialización**

* SolucionActual: Ídem algoritmo anterior.
* K: Ídem algoritmo anterior.
* Añadidos: Variables que ya han sido añadidas.
* Eliminados: Variables que han sido elimnadas

**Algoritmo**:

1. Mientras que no se cumpla condición de parada:
   1. Seleccionar y añadir la mejor variable V del conjunto original de variables que no se encuentre en SolucionActual ni Añadidos.
      1. SolucionTemporal = SolucionActual + V
      2. Evaluar SolucionTemporal y guardar su rendimiento.
   2. Seleccionar la mejor SoluciónTemporal y hacer SolucionActual = MejorSolucionTemporal.
   3. Actualizar Añadidos con la nueva variable.
   4. Seleccionar y añadir la peor variable del conjunto original de variables que no se encuentre en Eliminados.
      1. SolucionTemporal = SolucionActual - V
      2. Evaluar SolucionTemporal y guardar su rendimiento.
   5. Seleccionar la mejor SoluciónTemporal. Solo si el rendimiento de la mejor Solución temporal es superior al rendimiento de la mejor solucion

obtenida en el punto 2, entonces: SoluciónActual = MejorSolucionTemporal. En este caso, actualizar Eliminados añadiendo la variable eliminada.

* 1. Evaluar condición de parada.

1. Devolver tabla con las MejorSolucionTemporal.

# Resultados

Una vez implementada la metodología, incluyendo funciones y métodos implementados, desde el grupo de trabajo pasamos a realizar test y pruebas, en primer lugar para verificar la funcionalidad que se pretendía implementar, para a posteriori comenzar el trabajo de análisis de resultados con 2 conjuntos de datos independientes procesados con anterioridad, proporcionados en la propuesta de este trabajo, que permitirían lanzar hipótesis y realizar ciertas observaciones que consideramos de interés.

* Pruebas de procedimientos individuales: método de evaluación robusta.

En primer lugar, quisimos evaluar los resultados del método de evaluación robusta, basado en validación cruzada de K pliegues, como describimos anteriormente. Para ello, en primera instancia, decidimos pasar al método los conjuntos de datos con los que trabajamos, y un conjunto de variables concretas, de mayor a menor tamaño, de tal manera que nuestra hipótesis sería que, de manera sistemática, salvo ciertos casos que luego estudiaremos, el método debería ofrecer puntuaciones más elevadas, con los conjuntos más grandes de datos, más que con los más reducidos.

Como principio básico a la hora de realizar experimentación en la que vamos a comparar resultados, cabe mencionar que se han probado ambos conjuntos bajo las mismas propiedades de configuración a excepción del propio conjunto y de las variables que procesa cada uno de ellos. En concreto, se han obtenido ejecutando 20 experimentos para realizar el promedio de las soluciones, obtener un resultado con cierto grado de precisión y minimizar el efecto de la aleatoriedad que puede alterar los resultados. También, se han ejecutado 10 pliegues, que consideramos razonable para reducir la influencia de las diferencias que provocan realizar los experimentos con un conjunto de entrenamiento y pruebas, que sobre otro. Finalmente, hemos dejado los valores por defecto en lo que respecta al tipo de algoritmo de aprendizaje que pretendemos que use la función, árboles de clasificación binarios, así como la métrica de evaluación concreta.

Tras procesar los métodos, realizar las importaciones necesarias, procesar los ficheros fuentes de datos y finalmente ejecutar los métodos, a continuación, mostramos los resultados en estos dos conjuntos de datos.

* + Conjunto de datos: Titanic

En primer lugar, realizamos las pruebas con el conjunto de datos con los pasajeros del navío Titanic.

* Tabla 1. Evaluación del conjunto de datos ‘Titanic’ con diferentes conjuntos de variables

| Conjuntos de variables | Puntuación |
| --- | --- |
| 'Pclass','Sex','Age','SibSp','Parch','Fare','Embarked','Initial','Age\_band','Family\_Size','Alone','Fare\_cat','Deck','Title','Is\_Married' | 0.7684998726763425 |
| 'Embarked','Age\_band','Family\_Size' | 0.6190617661205894 |
| ‘Age\_band’ | 0.5403229776759183 |
| 'Initial' | 0.7833541295305997 |

* + Conjunto de datos: Tumores

A continuación, realizamos las pruebas con el conjunto de datos con los casos detectados de tumores.

* Tabla 2. Evaluación del conjunto de datos ‘Tumores’ con diferentes conjuntos de variables

| Conjuntos de variables | Puntuación |
| --- | --- |
| 'mean radius','mean texture','mean perimeter','mean area','mean smoothness','mean compactness','mean concavity','mean concave points','mean symmetry','mean fractal dimension','radius error','texture error','perimeter error','area error','smoothness error','compactness error','concavity error','concave points error','symmetry error','fractal dimension error','worst radius','worst texture','worst perimeter','worst area','worst smoothness','worst compactness','worst concavity','worst concave points','worst symmetry','worst fractal dimension' | 0.9100474386724382 |
| 'mean radius','mean texture','mean perimeter' | 0.8820790043290033 |
| 'mean radius' | 0.7916919191919186 |

* + Conclusiones y observaciones acerca de este experimento.

Tal y como realizamos la hipótesis que expusimos anteriormente, un hecho que observamos ejecutando ambos conjuntos de prueba, es que aunque no tenga por qué ocurrir así necesariamente como explicaremos posteriormente y podemos observar en un caso en el ejemplo del conjunto del Titanic, aunque de manera generalizada como podemos observar, tiende a tener una menor tasa de acierto cuando reducimos el número de variables, causado porque el entrenamiento, con un mayor número de variables, ofrecerá una mayor precisión a la hora de obtener un resultado cuando existe una relación directa entre variables y variable objetivo.

Esta variabilidad que presenta incluso en ocasiones con mayor tasa de error con un mayor número de variables es causada por la presencia de variables que introducen ruido en el entrenamiento y provocan una tasa de errores superior al ejecutar pruebas. Estos factores dependerán del conjunto de entrenamiento y de las variables que han sido seleccionadas. Un ejemplo claro, es el que podemos observar en la última prueba realizada al conjunto de datos Titanic, como se puede observar en la Tabla 1. Este fenómeno comprendemos que se produce porque ‘Initial’ en una variable que de por si sola, se ajusta mejor al resultado al final que el conjunto de soluciones anteriores formado por más variables, incluso cuando este contiene a la misma variable, ya que debemos recordar que no se evalúa una variable una por una, sobre su efectividad conjunta, que no tiene por qué ser mayor, como acabamos de comprobar.

Por otra parte, otro factor determinante a la hora de ejecutar esta función es el parámetro de pliegues que realiza la validación cruzada. La razón para incluirlo en este algoritmo reside en que los métodos de aprendizaje suelen incluir una componente de aleatoriedad. En este caso, buscamos regularizar el método, por lo que utilizamos este preciso procedimiento, junto con, a voluntad del usuario, introducir una variable para repetir el conjunto de los experimentos todas aquellas iteraciones que se desee para hacer una media de los resultados de cada una de ellas, y así obtener un resultado promedio que haga que el resultado sea más preciso, a costa de una complejidad mucho mayor, y un tiempo mayor. Esto no se refleja en un experimento tal cual, sino probando la ejecución en varias ocasiones, comprobando como con un mayor número de pliegues, más estables deben ser los datos obtenidos.

* Pruebas de procedimientos individuales: *Sequential Forward Selection*.

El siguiente caso de estudio, realizaremos pruebas sobre ambos conjuntos de datos a los métodos de búsqueda secuencial para determinar las mejores características y variables que podemos extraer del conjunto de variables que les vayamos introduciendo al método, esto es, los conjuntos que nos devuelvan una tasa de acierto superior.

El procedimiento para ejecutar las pruebas será el mismo que en los casos anteriores. Probaremos de un mayor conjunto con todas las variables, a otros de tamaño más reducido, para posteriormente realizar un análisis de los resultados. Con el fin de mantener unicidad y mismo criterio en los resultados entre todas las pruebas de este método, así como el resto de pruebas, se ha seleccionado el mismo algoritmo de entrenamiento y de aprendizaje, árboles binarios, y la misma métrica.

Tras procesar los métodos, realizar las importaciones necesarias, procesar los ficheros fuentes de datos y finalmente ejecutar los métodos, a continuación, mostramos los resultados en estos dos conjuntos de datos.

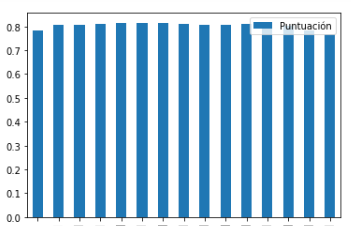
* + Conjunto de datos: Titanic

En primer lugar, realizamos las pruebas con el conjunto de datos con los pasajeros del navío Titanic.

* Tabla 3. Evaluación del conjunto de datos ‘Titanic’ con todas las variables

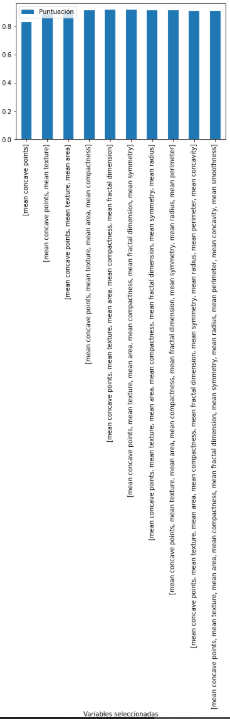
| Índice | Mejores características | | |
| --- | --- | --- | --- |
| Conjuntos de variables | Tamaño | Puntuación |
| 0 | [Initial] | 1 | 0.783354 |
| 1 | [Initial, SibSp] | 2 | 0.805218 |
| 2 | [Initial, SibSp, Deck] | 3 | 0.807720 |
| 3 | [Initial, SibSp, Deck, Fare\_cat] | 4 | 0.809612 |
| 4 | [Initial, SibSp, Deck, Fare\_cat, Title] | 5 | 0.815833 |
| 5 | [Initial, SibSp, Deck, Fare\_cat, Title, Sex] | 6 | 0.816190 |
| 6 | ['Initial', 'SibSp', 'Deck', 'Fare\_cat', 'Title', 'Sex', 'Is\_Married'] | 7 | 0.815761 |
| 7 | ['Initial','SibSp','Deck','Fare\_cat','Title','Sex', 'Is\_Married',  'Alone'] | 8 | 0.809593 |
| 8 | ['Initial','SibSp','Deck','Fare\_cat','Title','Sex', 'Is\_Married',  'Alone','Pclass'] | 9 | 0.808535 |
| 9 | ['Initial','SibSp','Deck','Fare\_cat','Title','Sex', 'Is\_Married',  'Alone','Pclass', ‘Fare’] | 10 | 0.804868 |
| 10 | ['Initial','SibSp','Deck','Fare\_cat','Title','Sex', 'Is\_Married',  'Alone','Pclass', ‘Fare’, ‘Family\_Size’] | 11 | 0.808924 |
| 11 | ['Initial','SibSp','Deck','Fare\_cat','Title','Sex', 'Is\_Married',  'Alone','Pclass', ‘Fare’, ‘Family\_Size’, ‘Parch’] | 12 | 0.809724 |
| 12 | ['Initial','SibSp','Deck','Fare\_cat','Title','Sex', 'Is\_Married',  'Alone','Pclass', ‘Fare’, ‘Family\_Size’, ‘Parch’, 'Embarked' ] | 13 | 0.804570 |
| 13 | ['Initial','SibSp','Deck','Fare\_cat','Title','Sex', 'Is\_Married',  'Alone','Pclass', ‘Fare’, ‘Family\_Size’, ‘Parch’, 'Embarked' , 'Age\_band'] | 14 | 0.782057 |
| 14 | ['Initial','SibSp','Deck','Fare\_cat','Title','Sex', 'Is\_Married',  'Alone','Pclass', ‘Fare’, ‘Family\_Size’, ‘Parch’, 'Embarked' , 'Age\_band', 'Age'] | 15 | 0.768691 |

Ofrecemos, además, una representación en escala de las puntuaciones obtenidas en cada conjunto, ordenadas el orden de la tabla.



1. . Tabla representativa con los valores de las puntuaciones según el conjunto de variables sobre el conjunto de datos ‘Titanic’. Elaboración propia.
   * Conjunto de datos: Tumores

A continuación, realizamos las pruebas con el conjunto de datos con los casos detectados de tumores, aunque con la finalidad de no exceder la longitud máxima del documento y no mostrar información redundante, en este caso mostraremos únicamente la gráfica estadística y en un número reducido de variables, donde en el pie, como el caso anterior, podrá observarse el conjunto de variables evaluado, y en la altura la puntuación obtenida. En el fichero notebook, no obstante, se puede consultar el resultado completo.



1. . Tabla representativa con los valores de las puntuaciones según un conjunto de variables reducido sobre el conjunto de datos ‘Tumores’. Elaboración propia.
   * Conclusiones y observaciones acerca de este experimento.

Como fenómeno a destacar en las pruebas podemos observar que la puntuación aumenta paulatinamente poco a poco conforme va añadiendo las mejores variables, para ir reduciendo su puntuación gradualmente con respecto se van introduciendo aquellas que son peores. Por otra parte, puede interpretarse como que este fenómeno aparece cuando se produce un sobre-entrenamiento debido a un exceso de variables, debido a que existen variables que introduzcan ruido o que, pese a realizar validación cruzada, existan casos sobre los que se realizan entrenamientos o evaluaciones erróneas porque existan casos puntuales en los conjuntos de entrenamiento que sean incorrectos, aunque en estas pruebas, contaremos como que no se produce esta casuística y son correctos.

Por lo tanto, el conjunto de variables con mejor puntuación corresponde a los valores intermedios de la tabla, que además reduce el coste al no contener el conjunto de variables entero; aunque debido a la dependencia de los resultados con respecto al conjunto de entrenamiento y pruebas, no nos permite descartar definitivamente el resto de variables, aunque sirven como orientación, y si deseamos reducir la complejidad de una función de aprendizaje, podríamos tomar el conjunto de variables que ofrezca un mejor promedio entre puntuación y tamaño.

* Pruebas de procedimientos individuales *Sequential Floating Forward Selection*.

Finalmente, tras probar el método de búsqueda secuencial hacia adelante anterior, probamos el método de búsqueda secuencial hacia adelante y atrás sobre los mismos conjuntos de datos y bajo las mismas condiciones y parámetros, para comprobar su funcionamiento, extraer conclusiones y reflexionar acerca de ambos algoritmos. Probaremos de un mayor conjunto con todas las variables, a otros de tamaño más reducido, para posteriormente realizar un análisis de los resultados. Con el fin de mantener unicidad y mismo criterio en los resultados entre todas las pruebas de este método, así como el resto de las pruebas, se ha seleccionado el mismo algoritmo de entrenamiento y de aprendizaje, árboles binarios, y la misma métrica.

Tras procesar los métodos, realizar las importaciones necesarias, procesar los ficheros fuentes de datos y finalmente ejecutar los métodos, a continuación, mostramos los resultados en estos dos conjuntos de datos.

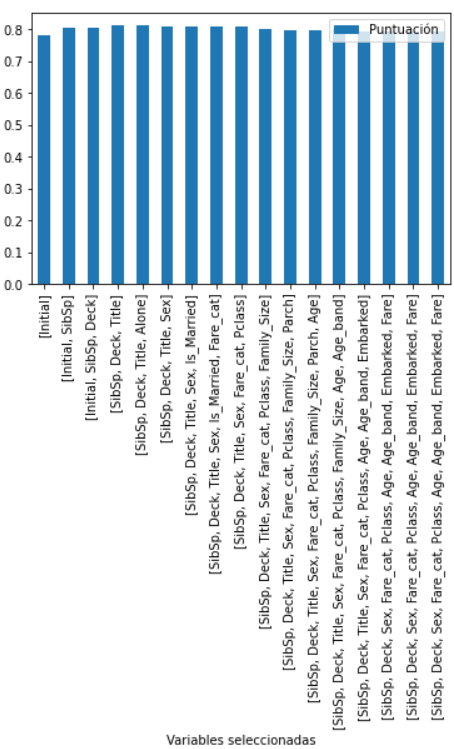
* + Conjunto de datos: Titanic

En primer lugar, realizamos las pruebas con el conjunto de datos con los pasajeros del navío Titanic.

* Tabla 4. Optimización del conjunto de variables‘Titanic’ con todas las variables

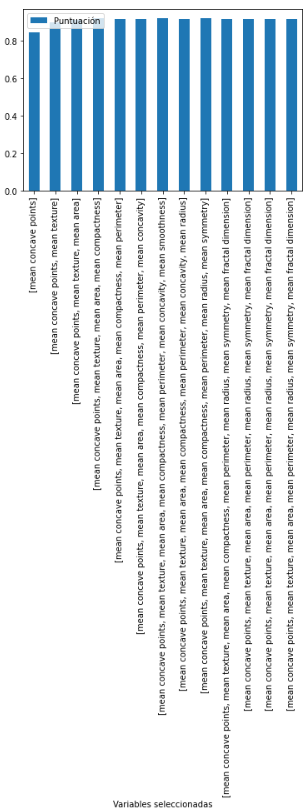
| Índice | Mejores características | | |
| --- | --- | --- | --- |
| Conjuntos de variables | Tamaño | Puntuación |
| 0 | [Initial] | 1 | 0.783354 |
| 1 | [Initial, SibSp] | 2 | 0.805218 |
| 2 | [Initial, SibSp, Deck] | 3 | 0.807720 |
| 3 | [SibSp, Deck, Title] | 3 | 0.812598 |
| 4 | [SibSp, Deck, Title, Alone] | 4 | 0.812229 |
| 5 | [SibSp, Deck, Title, Sex] | 4 | 0.811127 |
| 6 | [SibSp, Deck, Title, Sex, Is\_Married] | 5 | 0.811127 |
| 7 | [SibSp, Deck, Title, Sex, Is\_Married, Fare\_cat] | 6 | 0.810445 |
| 8 | [SibSp, Deck, Title, Sex, Fare\_cat, Pclass] | 6 | 0.8102545 |
| 9 | ['SibSp', 'Deck', 'Title', 'Sex', 'Fare\_cat', 'Pclass', 'Family\_Size'] | 7 | 0.802809 |
| 10 | ['SibSp', 'Deck', 'Title', 'Sex', 'Fare\_cat', 'Pclass', 'Family\_Size', 'Parch'] | 8 | 0.799357 |
| 11 | ['SibSp', 'Deck', 'Title', 'Sex', 'Fare\_cat', 'Pclass', 'Family\_Size', 'Parch', ‘Age’] | 9 | 0.798482 |
| 12 | ['SibSp', 'Deck', 'Title', 'Sex', 'Fare\_cat', 'Pclass', 'Family\_Size', 'Age', 'Age\_band'] | 9 | 0.800137 |
| 13 | ['SibSp', 'Deck', 'Title', 'Sex', 'Fare\_cat', 'Pclass', 'Age', 'Age\_band', 'Embarked'] | 9 | 0.792197 |
| 14 | ['SibSp', 'Deck', 'Sex', 'Fare\_cat', 'Pclass', 'Age', 'Age\_band', 'Embarked', 'Fare'] | 9 | 0.792800 |
| 15 | ['SibSp', 'Deck', 'Sex', 'Fare\_cat', 'Pclass', 'Age', 'Age\_band', 'Embarked', 'Fare'] | 9 | 0.792800 |
| 16 | ['SibSp', 'Deck', 'Sex', 'Fare\_cat', 'Pclass', 'Age', 'Age\_band', 'Embarked', 'Fare'] | 9 | 0.792800 |

Ofrecemos, además, una representación en escala de las puntuaciones obtenidas en cada conjunto, ordenadas el orden de la tabla.



1. . Tabla representativa con los valores de las puntuaciones según un conjunto de variables reducido sobre el conjunto de datos ‘Titanic. Elaboración propia.
   * Conjunto de datos: Tumores

A continuación, realizamos las pruebas con el conjunto de datos con los casos detectados de tumores, aunque con la finalidad de no exceder la longitud máxima del documento y no mostrar información redundante, en este caso mostraremos únicamente la gráfica estadística y en un número reducido de variables, donde en el pie, como el caso anterior, podrá observarse el conjunto de variables evaluado, y en la altura la puntuación obtenida. En el fichero notebook, no obstante, se puede consultar el resultado completo.

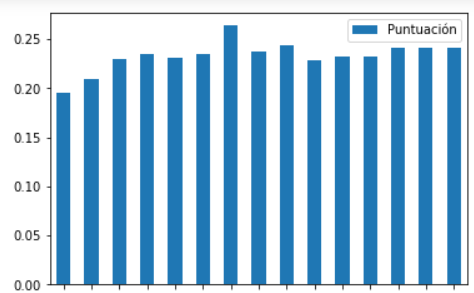


1. . Tabla representativa con los valores de las puntuaciones según un conjunto de variables reducido sobre el conjunto de datos ‘Tumores’. Elaboración propia.
   * Conclusiones y observaciones acerca de este experimento.

Experimentando con este método podemos observar un fenómeno similar al que ocurría en el anterior método. Observamos como las mejores soluciones no se encuentran en los casos en los que más tamaño registra, o últimas iteraciones del algoritmo, sino en aquellos en los que se encuentran las mejores soluciones que tienen un menor número de variables. Para seleccionar las mejores variables deberíamos elegir aquellas que nos ofrezcan una mejor relación entre tamaño del conjunto de variables y su tasa de aciertos. Sin embargo, mientras el anterior algoritmo comenzaba aumentando sus puntuaciones conforme comenzaban las iteraciones, registraba los mejores casos en los casos cercanos a la mitad de todas las iteraciones, y finalmente los últimos casos solía bajar esa puntuación, podemos observar como la progresión de la puntuación se estanca en las últimas iteraciones de este algoritmo, ya que aunque los mejores casos siguen localizándose en la mitad de todas las iteraciones, poco a poco se estabilizan al final del algoritmo en busca de la mejor combinación posible de variables, esto es fácilmente apreciable en las gráficas que generamos tras los experimentos.

* Pruebas adicionales con otros conjuntos de datos.

Con el fin de realizar un análisis más pormenorizado, hemos realizado más pruebas del algoritmo bajo casos de prueba planteados con otros conjuntos de datos, uno para problemas multi clase y otro para problemas binarios, tal y como los que se proporcionan como ejemplo adscritos a este trabajo. Como conclusión a dichos experimentos, pese a la diferencia en la naturaleza del problema que representan, los resultados se mantienen en la línea de los anteriores casos, en concreto, a continuación, mostramos el resultado de aplicar el algoritmo SFFS sobre un conjunto de prueba planteado como un problema multi clase por los resultados posibles de la variable objetivo. Una particularidad de este experimento es que, al tratarse de un problema multi clase, podemos observar como el número de acierto es mucho menor, al haber un mayor número de opciones posibles, lo que hace aumentar la probabilidad de fallo.

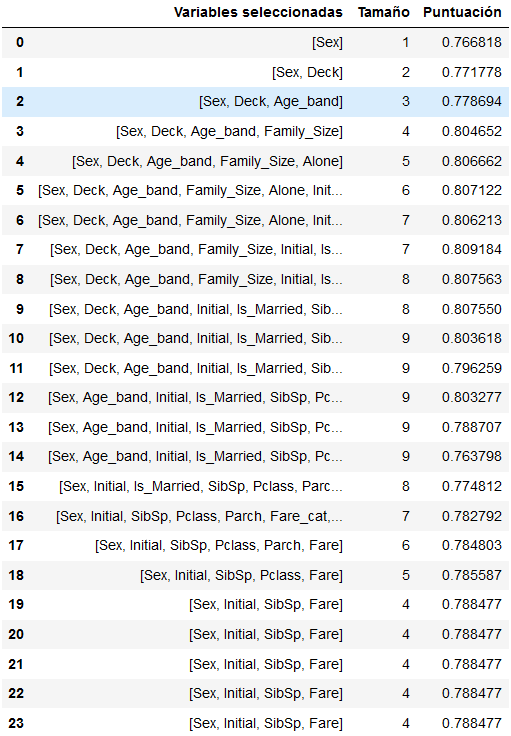


1. . Tabla representativa con los valores de las puntuaciones según un conjunto de variables reducido sobre el conjunto de datos ‘Vino’. Elaboración propia.

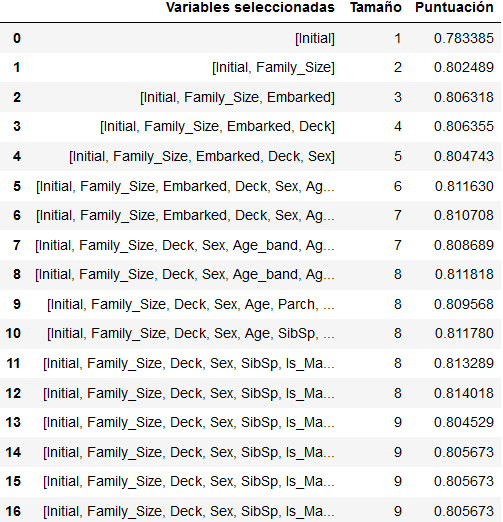
* Pruebas con respecto al estimador seleccionado

Realizando estas pruebas, pretendemos realizar un estudio del impacto que supone utilizar un estimador u otro a la hora de decidir qué algoritmo de estimación y clasificación deberíamos incluir por defecto, analizando su efectividad y rendimiento, teniendo en cuenta la alta complejidad que pueden suponer los algoritmos implementados.

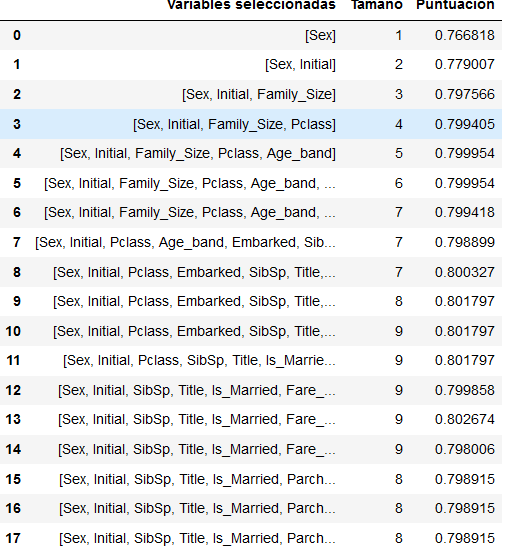
Los resultados son los que se muestran a continuación, corresponden a ‘k’ vecinos, redes neuronales y Naive Bayes, respectivamente.



1. . Tabla representativa con los valores de las puntuaciones sobre el conjunto de datos ‘Titanic’ del algoritmo SFFS, utilizando el algoritmo ‘k’ vecinos, hasta la solución final. Elaboración propia.



1. . Tabla representativa con los valores de las puntuaciones sobre el conjunto de datos ‘Titanic’ del algoritmo SFFS, utilizando el algoritmo de redes neuronales, hasta la solución final. Elaboración propia.



1. . Tabla representativa con los valores de las puntuaciones sobre el conjunto de datos ‘Titanic’ del algoritmo SFFS, utilizando el algoritmo Naïve Bayes, hasta la solución final. Elaboración propia.

Tras hacer un análisis de los resultados que ofrece ejecutar el algoritmo SFFS usando varios estimadores, extraemos conclusiones relativamente interesantes que nos pueden ayudar a tomar una decisión fundamentada acerca de cuál debería ser el algoritmo de clasificación por defecto que deberíamos integrar en al utilizar las pruebas.

En primer lugar, podemos apreciar fácilmente como tanto Naive Bayes como los 'k' vecinos presentan una tasa de acierto más reducida de media, mientras que en segundo lugar podemos apreciar como la estimación basada en árboles se encuentra en una segunda posición tras el algoritmo de redes neuronales, que aparenta ser el más efectivo.

Sin embargo, no debemos de pasar por alto el tamaño del conjunto en el que converge el método en cada caso. Los dos algoritmos que ofrecen una mayor precisión y puntuación son aquellos que convergen en conjuntos de elementos de tamaño más elevado, seguido de cerca por Naive Bayes, mientras que los 'k' vecinos, pese a ser de los menos precisos, ofrece un conjunto de la mitad de tamaño del anterior, lo que no debe ser pasado por alto.

En lo que concierne la complejidad de los algoritmos, ofrecen un rendimiento equilibrado la mayor parte de estos algoritmos, pero debemos destacar negativamente el rendimiento que ha ofrecido el algoritmo basado en redes neuronales, que ha sido en varias órdenes de longitud más lento que el resto, lo que podría ser solucionado optimizando los parámetros como los de penalización, número de capas ocultas, entre otros a costa de la precisión.

Como conclusión, apunto que el método por defecto, de árboles de decisión, es el más equilibrado, ya que ofrece tasas de acierto consistentes y elevadas con un coste computacional medio.

# Conclusiones

En este apartado del documento, hemos visto conveniente expresar nuestro punto de vista y opinión fundada en el estudio teórico que hemos tenido que llevar a cabo para desarrollar este trabajo, los fenómenos y experimentos que hemos llevado a cabo en la implementación y pruebas que hemos realizado y llevado a cabo durante este estudio de investigación.

Mediante la implementación de este algoritmo, creo que hemos conseguido el objetivo de este proyecto, y es poder seleccionar las mejores variables, o aquellas que presenten una mayor representatividad sobre el resultado final, y que puedan ser utilizados para aplicar en procedimientos de Machine Learning para reducir el coste computacional y complejidad de los mismos al tratar con información relacionada al conjunto de datos sobre el que se ha realizado la optimización.

Tras toda la fase de pruebas y experimentación, hemos concluido que, aunque el coste computacional es varias órdenes superior en caso del algoritmo *Sequential Floating Forward Selection,* los resultados obtenidos utilizando este algoritmo son mejores que los obtenidos con el algoritmo *Sequential Forward Selection,* porque aunque en ambos cuenten con las mejores soluciones en mitad de todas las iteraciones, las soluciones que aporta el primer algoritmo son de un tamaño más reducido, con las ventajas que ello conlleva, tienen una media superior en conjunto con respecto a la del segundo algoritmo, las valoraciones más altas son encontradas en el primero, y aunque no sea un factor tan relevante, hay un mayor número de soluciones.

La razón de tan enorme cantidad de ventajas comparando un algoritmo u otro radican en que el segundo presenta una mayor flexibilidad a la hora de gestionar las soluciones que se van a devolver como respuesta al método, con la posibilidad de probar combinaciones que no pueden darse por el orden de las variables, y por poder eliminar variables, sin forzar a contenerlas todas.

En caso de elegir entre un algoritmo u otro, en nuestra opinión, deberíamos atender al contexto en el que se trabaja, ya que un algoritmo de mayor complejidad puede no compensar en ser utilizado en casos muy puntuales, ya que la diferencia entre los resultados entre ambos métodos no sea tan relevante.

El gran problema que presentan estos algoritmos, y que juega en favor del primer algoritmo, radica de forma inevitable en el coste computacional que conllevan, muy superior a otros algoritmos similares. Desde el grupo de trabajo hemos tratado de optimizar el rendimiento del algoritmo mediante la inclusión de mecanismos de paralelización que aprovechan mejor la capacidad del equipo sobre el que se ejecutan, sin embargo, sigue teniendo un coste computacional muy elevado, por lo que como sugerencia de trabajo futuro, sugeriría que se estudiase el mismo problema enfocado de una idea de base diferente, y es utilizando un algoritmo de búsqueda metaheurístico o de otra naturaleza que permita, aunque introduzca alguna restricción, optimizar el coste computacional.

A nivel personal, este trabajo nos ha ayudado a ampliar conocimientos a un plano más cercano a la realidad acerca de la Inteligencia Artificial y como toma de contacto a la misma, lo que consideramos que nos ha enriquecido.

##### Referencias

1. Página web del curso IA de Ingeniería del Software. [https://www.cs.us.es/cursos/iais. Consultada el 01/04/2020](https://www.cs.us.es/cursos/iais.%20Consultada%20el%2001/04/2020).
2. “Las cifras de la Inteligencia Artificial en el futuro” de.la publicación eldiario.es <https://www.eldiario.es/tecnologia/cifras-Inteligencia-Artificial-futuro_0_644985749.html>. Consultada el 02/05/2020.
3. “Aprendizaje basado en árboles de decisión” en Wikipedia <https://es.wikipedia.org/wiki/Aprendizaje_basado_en_%C3%A1rboles_de_decisi%C3%B3n> Consultada el 02/05/2020
4. “A Beginner’s Guide to Classification and Regression Trees” <https://www.digitalvidya.com/blog/classification-and-regression-trees/> Consultada el 03/05/2020
5. “Data Mining con Árboles de Decisión”, por Jorge Martín Arevalillo, <https://web.fdi.ucm.es/posgrado/conferencias/JorgeMartin-slides.pdf> Consultada el 03/05/2020
6. “Regresión: selección de variables. Stepwise, Forward, Backward” <https://statisticalecology-ec.blogspot.com/2012/08/regresion-seleccion-de-variables.html> Consultada el 03/05/2020
7. “¿Cómo seleccionar las variables adecuadas para tu modelo?” <https://www.maximaformacion.es/blog-dat/como-seleccionar-las-variables-adecuadas-para-tu-modelo/> Consultada el 03/05/2020
8. “Estructuras de Datos” - <https://www.scoop.it/topic/estructura-de-datos/p/3692056961/2012/12/12/6-1-busqueda-secuencial> Consultada el 03/05/2020
9. “Sequential Feature Selection” - <https://es.mathworks.com/help/stats/sequential-feature-selection.html> Consultada el 03/05/2020