CURSO MODULAR

BIG DATA Y DATA SCIENCE APLICADOS A LA ECONOMÍA Y A LA ADMINISTRACIÓN Y DIRECCIÓN DE EMPRESAS

| Modulo: MINERÍA DE DATOS II

Autores:

Pablo Sánchez, Angel Rodríguez y Alfonso Carabantes

Organizadores

UNED

aeca

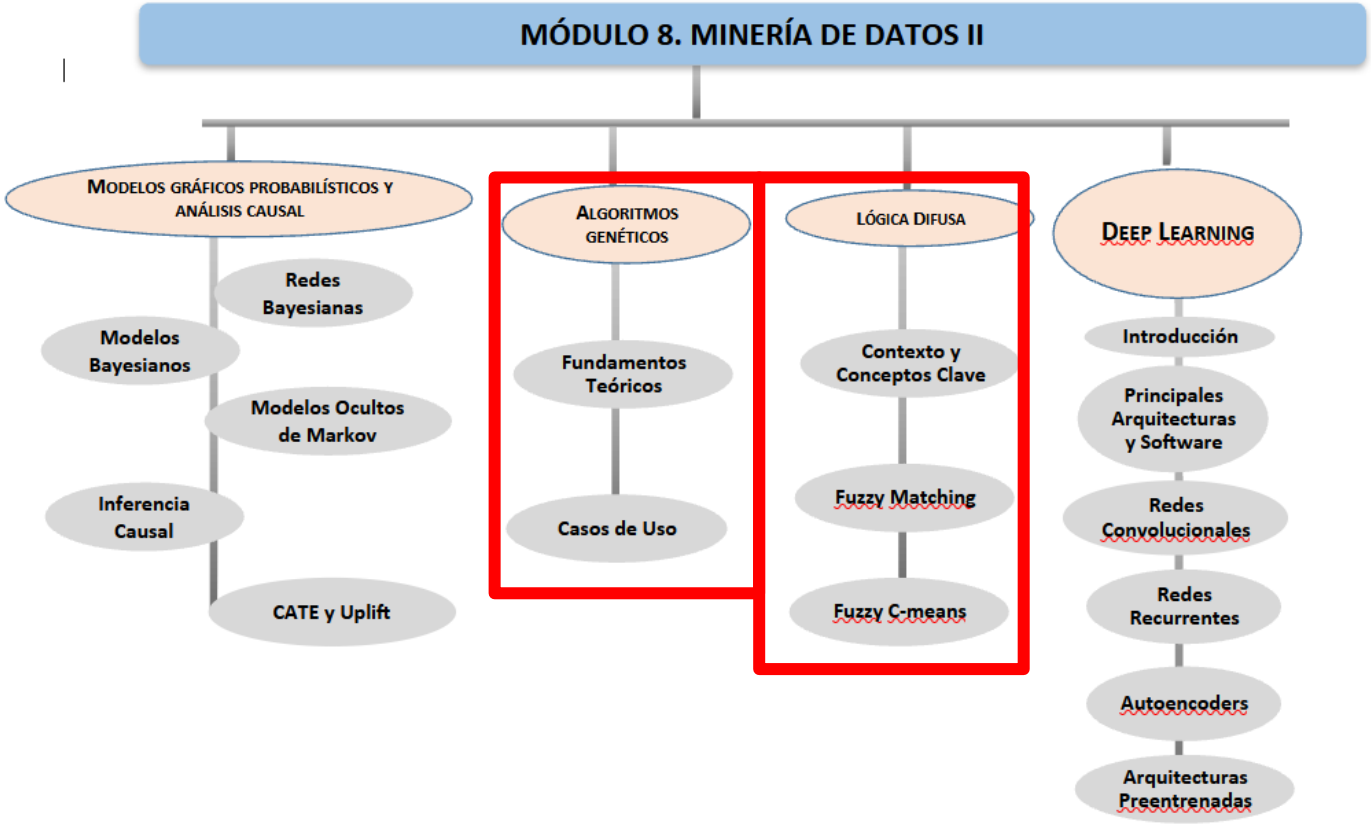
Asociación Española de Contabilidad y Administración de Empresas

Colaborador

Cuarta parte

15 de julio de 2024

MINERÍA DE DATOS II



ÍNDICE

- Algoritmos Genéticos
  - Fundamentos teóricos
  - Casos de uso
  - Ejemplos en Python
- Lógica Difusa
  - Conceptos básicos
  - Operaciones: T-Normas y T-Conormas
  - Funciones de membresía y modelado difuso
  - Ejemplos en lógica difusa

ALGORITMOS GENÉTICOS-INTRODUCCIÓN

Los **Algoritmos Genéticos o Evolutivos**, se inspiran en el proceso natural de selección y evolución, tal y como se describe por la **teoría evolucionista de la selección natural** postulada por **Darwin**.
Los **principios** sobre los que se asientan los algoritmos evolutivos y, en concreto, los **genéticos** son:

- Los individuos **mejor adaptados** al entorno son aquellos que tienen una probabilidad **mayor** de **sobrevivir** y, por ende, de **reproducirse**.

- Los **descendientes heredan** características de sus **progenitores**.

- De forma **esporádica** y natural se producen **mutaciones** en el **material genético** de algunos individuos, provocando cambios permanentes.

Los algoritmos genéticos son adecuados para obtener buenas aproximaciones en **problemas** de **búsqueda**, **aprendizaje** y **optimización** [Marczyk. 2004].

ALGORITMOS GENÉTICOS-INTRODUCCIÓN

Un **algoritmo genético** es una **función matemática** que, tomando como entrada unos **individuos iniciales** (población origen), **selecciona** aquellos **ejemplares** que **recombinándose** por algún método generarán como resultado la **siguiente generación**.

Esta función se aplicará de forma **iterativa** hasta verificar alguna **condición** de **parada**, bien pueda ser un número máximo de iteraciones o bien la obtención de un individuo que cumpla unas restricciones iniciales.

Los **algoritmos genéticos** fueron propuestos por **Holland**, quién intentando simular los procesos naturales de adaptación desarrolló por primera vez la idea de los **algoritmos genéticos en los años 60**. No obstante, no fue hasta 15 años más tarde cuando un **pupilo** suyo, **David Goldberg** los **aplicó** por **primera vez** a un problema real y los popularizó.

En 1985 se creó la primera conferencia mundial de algoritmos genéticos ICGA que se celebra hasta el día de hoy bianualmente.

ALGORITMOS GENÉTICOS-CONDICIONES APLICACIÓN

Condiciones de Aplicación

No es posible la aplicación en toda clase de problemas. Para que estos puedan aplicarse, los **problemas** deben **cumplir** las siguientes condiciones:

- El **espacio de búsqueda** debe estar acotado, por tanto ser **finito**.

- Es necesario poseer una **función de aptitud**, que denominaremos **fitness**, que **evalúe** cada solución (individuo) indicándonos de forma **cuantitativa** cuán **buena o mala** es una solución concreta.

- Las **soluciones** deben ser **codificables** en un lenguaje comprensible para un ordenador, y si es posible de la forma más compacta y abreviada posible.

Habitualmente, la segunda condición es la más complicada de conseguir y, habitualmente, se realizan conjeturas evaluándose los algoritmos con varias funciones de fitness.

ALGORITMOS GENÉTICOS-VENTAJAS

Ventajas

- **No necesitan ningún conocimiento particular del problema sobre el que trabajan**, únicamente cada ejemplar debe representar una posible solución al problema.

- Son **algoritmos admisibles**, es decir, con un número de **iteraciones** suficiente son capaces de obtener la **solución óptima** en problemas de optimización. **Teorema del Esquema de Holland** proporciona el fundamento teórico de porqué los Algoritmos Genéticos pueden resolver problemas.

- Los algoritmos genéticos son **bastante robustos frente a falsas soluciones**, ya que al realizar una inspección del espacio solución de forma **no lineal** (por ejemplo, si quisiéramos obtener el máximo absoluto de una función) el algoritmo no recorre la función de forma consecutiva, por lo que no se ve afectado por máximos locales.

- **Altamente paralelizables** (es decir, ya que el cálculo no es lineal podemos utilizar varias máquinas para ejecutar el programa y evaluar así un mayor número de casos).

- Pueden ser **incrustrables** en muchos algoritmos de data mining para formar modelos híbridos. Por ejemplo, para seleccionar el número óptimo de neuronas en un modelo de Perceptrón Multicapa o realizar la selección de variables en un modelo.

ALGORITMOS GENÉTICOS-INCONVENIENTES

Inconvenientes

- Su **coste computacional** puede llegar a ser muy **elevado**, si el espacio de trabajo es muy grande.

- En el caso de que **no** se haga un **correcto ajuste** de los **parámetros** pueden llegar a caer en una situación de dominación en la que se produce un **bucle infinito**, ya que unos individuos dominan sobre los demás impidiendo la evolución de la población y, por tanto, inhiben la diversidad biológica.

- Puede llegar a ser muy **complicado encontrar** una **función** de **evaluación** de cada uno de los individuos para seleccionar los mejores de los peores.

ALGORITMOS GENÉTICOS-CODIFICACIÓN

Codificación de los datos

Cada **individuo** o cromosoma está **formado** por unos cuantos **genes**. Para nuestro caso vamos a establecer que los individuos tienen un único cromosoma con una cierta cantidad de genes. Estos genes los consideramos como la cantidad mínima de información que se puede transferir. Los genes se pueden agrupar en características o rasgos que nos podrían ayudar en la resolución de ciertos problemas.

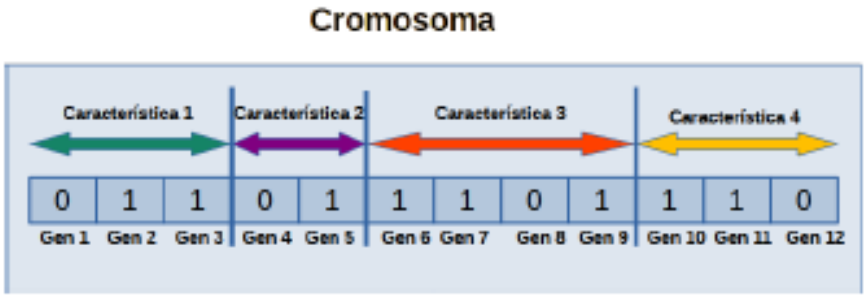
Estos **individuos** con sus **genes** los tenemos que **representar** de forma que podamos **codificar** esa **información**.

Los principales **métodos** de **representación** son:

- **Binaria:** Los individuos/cromosomas están representados por una serie de genes que son bits (valores 0 ó 1).
- **Entera:** Los individuos/cromosomas están representados por una serie de genes que son números enteros.
- **Real:** Los individuos/cromosomas están representados por una serie de genes que son números reales en coma flotante.
- **Permutacional:** Los individuos/cromosomas están representados por una serie de genes que son permutaciones de un conjunto de elementos. Se usan en aquellos problemas en los que la secuencia u orden es importante.
- **Basada en árboles:** Los individuos/cromosomas están representados por una serie de genes que son estructuras jerárquicas.

ALGORITMOS GENÉTICOS-CODIFICACIÓN

Codificación de los datos



Representación Binaria

0	1	1	0	1	1	1	0	1	1	1	0

Representación Entera

4	6	7	9	51	21	34	6	67	8	23	82

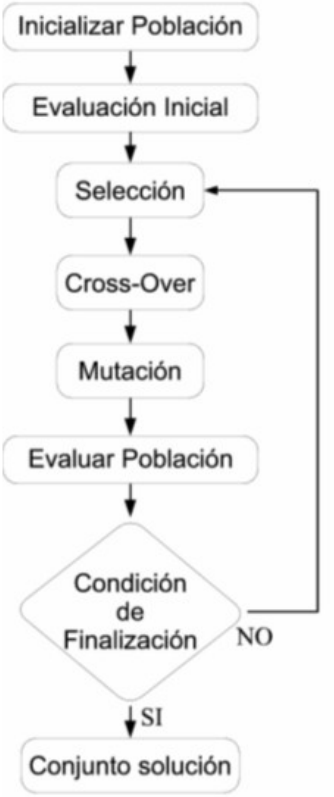
Representación Real

0,34	2,26	3,45	5,34	7,25	0,73	3,24	0,81	1,13	3,23	7,20	7,06

ALGORITMOS GENÉTICOS-ALGORITMO

Algoritmo

- Generar de forma **aleatoria** una serie de individuos (**Población Inicial**)
- **Evaluar** Población Inicial, obteniendo el **fitness** de cada individuo
- Iteración (condición de terminación es falsa).
  - Realizar la **selección** (los mejores se reproducen)
  - Realizar el **cross-over** (recombinación o reproducción)
  - Realizar **mutación**
  - **Evaluar** Población, obteniendo el **fitness** de cada individuo
- **Población final**



ALGORITMOS GENÉTICOS-PASOS

Población Inicial

- Habitualmente la **inicialización** se hace de forma **aleatoria** procurando una **distribución homogénea** en los casos iniciales de prueba. No obstante, si se tiene un **conocimiento** más profundo del **problema** es posible obtener mejores resultados **inicializando** la **población** de una **forma apropiada** a la clase de soluciones que se esperan obtener.

Evaluar Población

- Durante cada iteración (generación) cada **individuo** se **decodifica** convirtiéndose en un grupo de **parámetros** del **problema** y se **evalúa** el problema con esos datos con la **función fitness**..

ALGORITMOS GENÉTICOS-PASOS

Selección

La fase de **selección** **elige** los **individuos** a reproducirse en la próxima generación, esta selección puede realizarse por muy distintos métodos:

- **Selección elitista:** Se seleccionan los individuos con **mayor fitness** de cada generación. La mayoría de los algoritmos genéticos no aplican un elitismo puro sino que en cada generación evalúan el fitness de cada uno de los individuos, en el caso de que los mejores de la anterior generación sean mejores que los de la actual éstos se copian sin recombinación a la siguiente generación.

- **Selección proporcional a la aptitud:** los individuos **más aptos** (mayor fitness) tienen **más probabilidad** de ser seleccionados, asignándoles una probabilidad de selección más alta. Una vez seleccionadas las probabilidades de selección a cada uno de los individuos se genera una nueva población teniendo en cuenta éstas.

- **Selección por rueda de ruleta:** Es un método conceptualmente similar al anterior. Se le asigna una probabilidad absoluta de aparición de cada individuo de acuerdo al fitness de forma que ocupe un tramo del intervalo total de probabilidad (de 0 a 1). Una vez completado el tramo total se generan números aleatorios de 0 a 1 de forma que se seleccionen los individuos que serán el caldo de cultivo de la siguiente generación.

- **Selección por torneo**, **Selección por rango, Selección generacional, Selección por estado estacionario, Búsqueda del estado estacionario, Selección jerárquica**

ALGORITMOS GENÉTICOS-PASOS

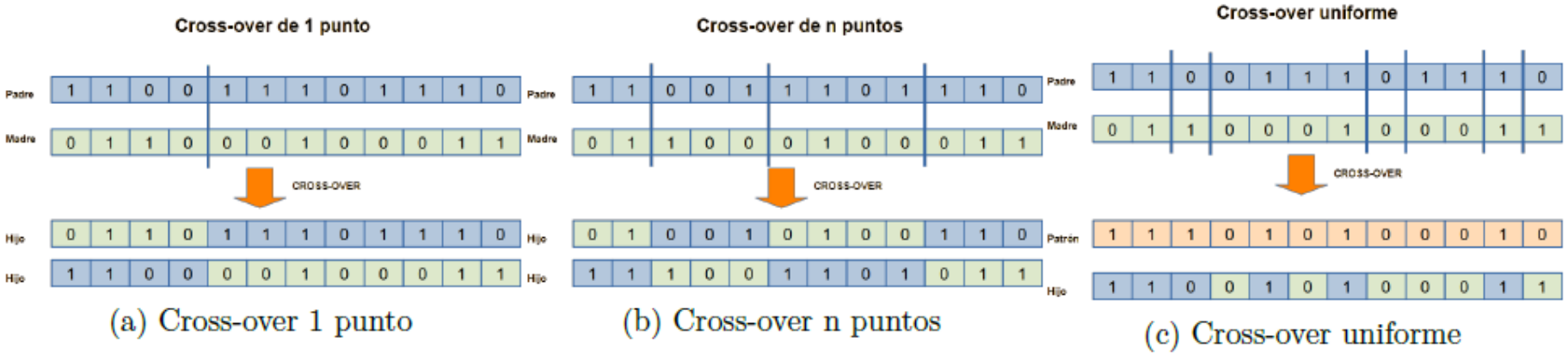
Cross-over (Recombinación o Reproducción)

La **recombinación** es el operador genético más utilizado y consiste en el **intercambio** de **material genético** entre dos elementos al azar (pueden ser incluso entre el mismo elemento). El material genético se intercambia entre bloques. Gracias a la **presión** selectiva irán predominando los **mejores bloques génicos**.
Existen diversos **tipos** de **cross-over**:

- **Cross-over de 1-punto.** Los cromosomas se cortan por 1 punto y el resultado se intercambia.
- **Cross-over de n-puntos.** Los cromosomas se cortan por n puntos y el resultado se intercambia.
- **Cross-over uniforme.** Se genera un patrón aleatorio en binario, y en los elementos que haya un 1 se realiza intercambio genético.
- **Cross-over especializados.** En ocasiones, el espacio de soluciones no es continuo y hay soluciones que a pesar de que sean factibles de producirse en el gen no lo son en la realidad, por lo que hay que incluir restricciones al realizar la recombinación que impidan la aparición de algunas combinaciones.

ALGORITMOS GENÉTICOS-PASOS

Cross-over (Recombinación o Reproducción)



(a) Cross-over 1 punto (b) Cross-over n puntos (c) Cross-over uniforme

ALGORITMOS GENÉTICOS-PASOS

Mutación

Las **mutaciones** son un mecanismo muy interesante por el cual es posible **generar nuevos individuos** con **rasgos distintos a sus predecesores**.
Tipos de mutaciones más comunes:

- **Mutación de gen:** existe una probabilidad de cambiar de valor un gen dentro del individuo.
- **Mutación multigen:** existe una probabilidad de cambiar de valor un conjunto de genes dentro del individuo.
- **Mutación de intercambio:** existe una probabilidad de intercambiar el contenido de dos genes aleatoriamente.
- **Mutación de barajado:** existe una probabilidad de que se produzca una mutación. De producirse, toma dos dos genes aleatoriamente y baraja de forma aleatoria los genes, según hubiéramos escogido, comprendidos entre los dos.

ALGORITMOS GENÉTICOS-PASOS

Mutación

Imagen 3.5: Mutacion gen

Imagen 3.6: Mutacion multigen

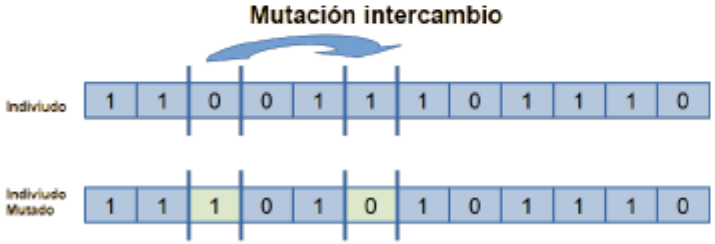


Imagen 3.7: Mutacion intercambio

Imagen 3.8: Mutacion barajado

ALGORITMOS GENÉTICOS-PASOS

Condición de Finalización

Una vez que se ha generado la nueva población se evalúa la misma y se selecciona a aquel individuo o aquellos que por su fitness se consideran los más aptos.

Seleccionados éstos se toman y evalúan, y si satisfacen **la condición de terminación** finaliza el algoritmo. Esta condición de parada será un valor que nosotros podamos considerar bueno normalmente asociado al **fitness** de la **mejor solución encontrada**.

Finalización por alcanzar el **máximo número** de **iteraciones**.

ALGORITMOS GENÉTICOS-PARAMETROS

Parámetros de un Algoritmo Genético

- Función fitness (fitness_func)
- Tipo de gen (gene_type)
- Número de generaciones (num_generations)
- Condición de parada específica (stop_criteria)
- Número de soluciones/individuos en la población (sol_per_pop)
- Número de soluciones seleccionadas como padres (num_parents_mating)
- Número de genes del cromosoma (num_genes)

ALGORITMOS GENÉTICOS-CASO DE USO

CASO DE USO DE SELECCIÓN DE OPTIMIZACIÓN DE FUNCIONES

El objetivo de este ejemplo es ver cómo podemos usar un **Algoritmo Genético** para calcular los parámetros de una función que queremos optimizar.

Supongamos que tenemos la función: $f(w_1, w_2, w_3, w_4, w_5, w_6) = 3w_1 - 1w_2 + 5w_3 + 5.2w_4 - 2w_5 - 4.3w_6$

donde los wi son parámetros de la función, y queremos aproximar esta función al valor 23.

Es decir, queremos encontrar los valores de los wi que al aplicarlos a la función se acercan al valor 23.

Usaremos un **Algoritmo Genético** con una **codificación** de **tipo real** (valores decimales), con un número de **6 genes**.

La **Función Fitness** nos tendrá que permitir **maximizar** el valor con lo cual la definiremos como: $fitness = \frac{1}{f(W) - 23 + 0.000001}$

Es decir, calculamos la diferencia entre el valor de la función con los parámetros y el 23, le añadimos 0.000001 (para evitar que dividamos por 0) y calculamos la inversa.
De esta forma cuanto más nos aproximemos al 23 más grande será el valor del fitness.

ALGORITMOS GENÉTICOS – EJEMPLOS EN PYTHON

Vamos a ver el Código en python con Keras/Tensorflow

ALGORITMOS GENÉTICOS-CASO DE USO

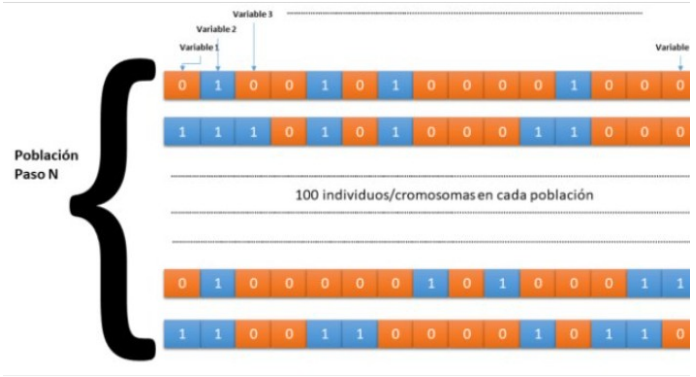
CASO DE USO DE SELECCIÓN DE VARIABLES

El objetivo de este ejemplo es ver cómo podemos usar un **Algoritmo Genético** para hacer una **selección de variables**, quedándonos sólo con unas pocas.

Vamos a trabajar con un **dataset** que clasifica información sobre **cáncer de mama.** Es un dataset con **30 variables** explicativas que nos dice si corresponde con cáncer o no cada entrada.

Lo primero a decidir de cara a utilizar el algoritmo genetico, será la **codificación** a usar en la definición de nuestro **cromosoma o individuo**. En nuestro caso será un vector de tamaño 30 (**30 genes**), que representa las 30 variables del dataset que hemos preparado. **Valor 0** significa que **no se usa esa variable**, **Valor 1** significa que **si se usa esa variable**.

En este caso la **Función Fitness** la construiremos teniendo como entrada cada **individuo** los **genes** que representan las **variables** que se van a usar. Tomaremos los **datos** sólo de **estas variables**, construiremos y **entrenaremos** un **Clasificador de Regresión Logística** y obtendremos el **Accuracy** conseguido con el entrenamiento. Este valor será nuestro **Fitness** para cada individuo (cada subconjunto de variables que probamos)



Cuanto **mayor** sea el **fitness** es porque cada vez es **mayor** el **Accuracy** del modelo.

ALGORITMOS GENÉTICOS – EJEMPLOS EN PYTHON

Vamos a ver el Código en python con Keras/Tensorflow

CURSO MODULAR

BIG DATA Y DATA SCIENCE APLICADOS A LA ECONOMÍA Y A LA ADMINISTRACIÓN Y DIRECCIÓN DE EMPRESAS

| Modulo: MINERÍA DE DATOS II

Autores:

Mauricio Beltrán, Pablo Sánchez y Alfonso Carabantes

Organizadores

UNED

aeca
Asociación Española de Contabilidad y Administración de Empresas

Colaborador

GRUPO DE USUARIOS DE R DE MADRID

Gracias por tu Atención!