MÉTODOS DE APRENDIZAJE AUTOMÁTICO PARA PREDECIR LA OBTENCIÓN DE ENERGIA EÓLICA

José M. Benjumeda Rubio

September 2020

1 Introducción

En este documento se hará una introducción al aprendizaje automático, explicando cuestiones generales y distintos métodos, para luego profundizar en métodos de regresión y redes neuronales. Se documentarán también pruebas a modo de aprendizaje, hasta finalmente construir una máquina de ensamble que utilice estos dos últimos métodos para realizar predicciones sobre la cantidad de energía eólica que se espera obtener en un parque.

1.1 Aprendizaje automático. Definición, concepto general y distintos tipos.

El aprendizaje automático es un método englobado dentro de "Data Science" o "Ciencia de Datos", donde se busca que una máquina realice operaciones cognitivas, que son las actividades, inicialmente propias del ser humano, que están relacionadas con el procesamiento de la información mediante el uso de la memoria, la atención, la percepción, la creatividad y el pensamiento abstracto o analógico.

Para conseguir que una máquina aprenda a realizar dichas tareas, se crea un modelo, que "aprende" con un conjunto de datos. Este aprendizaje consiste, a grandes rasgos, en ver qué respuesta va asociada a qué estímulo o entrada. Se busca que el modelo sea capaz de responder a estímulos que nunca había recibido, basándose en los patrones que ha observado en los datos con los que se entrenó.

Respecto al modelo, que junto con sus parámetros de configuración constituye la "máquina", no hay un modelo perfecto que funcione en todos los casos. La manera de construirlo siempre depende del problema que estemos tratando. A día de hoy, hay desarrollados más de 700 algoritmos para realizar estadísticas

sobre los datos, y cual de ellos o qué combinación de ellos utilizar es uno de los principales retos a superar para hacer aprendizaje automático. Aquí surge la primera gran clasificación de modelos según el tipo de aprendizaje, diferenciando tres categorías:

- Aprendizaje automático supervisado
- Aprendizaje automático no supervisado
- Aprendizaje automático reforzado

1.2 Aprendizaje automático supervisado, aprendizaje automático supervisado y aprendizaje automático reforzado

1.2.1 Aprendizaje automático supervisado.

Lo primero es decir que los datos se estructuran como un vector X que contiene todas las características excepto la característica considerara valor o resultado, y un vector de un solo elemento Y, que contiene éste último valor o resultado, que se considera dependiente de X. Con esto, el modelo intenta encontrar la función que relaciona esos valores de X con los de Y, para poder obtener los valores de Y para nuevas X que no existían en los datos de los que se disponía inicialmente. El modelo, una vez creado en su versión inicial, se entrena y mejora con los datos del conjunto de entrenamiento, y luego se pone a prueba con los datos del conjunto de test. Algunos ejemplos de algoritmos de aprendizaje automático supervisado son análisis de regresión, árboles de decisión, k vecinos más próximos, redes neuronales y máquinas de vectores de soporte.

1.2.2 Aprendizaje automático no supervisado.

Las principales diferencias entre este tipo de algoritmos y los supervisados es que en estos, los datos no tienen una etiqueta. Aquí se estudian y aprenden las relaciones entre los datos según sus características, sin hablar en ningún momento de un atributo etiqueta. Esto nos lleva a otra diferencia importante: en aprendizaje automático no supervisado no hay una forma directa de comprobar el porcentaje de error de una predicción, o cómo de buena ha sido. Los problemas a los que prestamos atención aquí son por ejemplo en cuántos grupos vamos a clasificar los datos, qué características van a ser más importantes y cuáles menos, etc.

En algunos casos no hay etiquetas inicialmente, pero en el futuro sí las habrá, y es entonces cuando nos conviene tener las observaciones ya clasificadas en grupos, debido a la posibilidad de que observaciones que pertenecen al mismo grupo (están relacionadas) puedan tener la misma etiqueta. El objetivo es que los grupos se hayan dividido según la etiqueta, pero antes de conocer ésta. Podemos pensar por ejemplo en un problema de clasificación de perfiles de una

red social, para que en cuanto se detecte un perfil falso, comprobar perfiles parecidos.

Más adelante hablaremos de algunos algoritmos, **COMPROBAR: como k-means clustering y vecinos más próximos.**

1.2.3 Aprendizaje automático reforzado

En los modelos de aprendizaje automático reforzado hay tres elementos principales: el agente, el entorno y un método de evaluar estados. El proceso consiste en que el agente conoce todas las posibles acciones que puede llevar a cabo, y, para cada estado, reparte una probabilidad inicialmente aleatoria a cada acción. Cada una de éstas, además, produce un cambio en el entorno. Podemos decir que el entorno tiene distintos estados, y en cada uno conviene más una acción u otra. Esto es lo que queremos aprender.

Paso por paso, lo que ocurre es lo siguiente:

1. El agente recibe el estado inicial.
2. El agente lleva a cabo una acción, elegida al azar, pero teniendo cada acción una probabilidad distinta.
3. El entorno cambia a otro estado
4. Si el estado actual es evaluable, se evalúa y se ajustan las posibilidades, si no, volvemos al paso 2.

1.3 Previo al aprendizaje. Aspectos importantes sobre los datos de partida.

La forma que tengan los datos que vamos a utilizar para que el modelo aprenda y lo que hagamos con ellos tiene un impacto directo en la calidad del modelo. Existen técnicas para mejorar los datos según el uso que les vayamos a dar, cambiçándoles el formato, el orden, dividiéndolos en distintos grupos, etc. A continuación explicamos algunas.

Es importante también que se hace un reparto del conjunto de datos en dos subconjuntos: conjunto de entrenamiento y conjunto de test. Con el conjunto de entrenamiento se perfila el modelo y se le añaden modificaciones, y luego se comprueba sobre el conjunto de test que funciona correctamente. Si no se obtienen los resultados deseados se cambia el modelo y se vuelve a empezar. Una vez satisfecho con los resultados, el modelo está listo para actuar con nuevos datos.

1.3.1 Depuración de datos (Scrubbing)

La depuración de datos un proceso que se hace cuando somos capaces de distinguir en el conjunto de datos, observaciones que se salen de lo habitual, de una manera que en ningún momento pretendemos ser capaces de predecir, o características que no parezcan estar relacionadas con el patrón que esperamos que nuestro modelo aprenda. Suele ocurrir cuando el conjunto de datos que tenemos es muy grande, pues el tamaño de éste y la cantidad de ruido a menudo son proporcionales, y uno de los procedimientos sería simplemente en eliminar de los datos o atributos que no nos aportan información útil, como casos muy excepcionales que es difícil que se repitan, por ejemplo. También puede resultar conveniente unir observaciones, pero hay que tener en cuenta que, mientras que al diferir en un campo con valor numérico, se pueden unir mediante la media, cuando son atributos que no son numéricos, como Japón y Corea del Sur, ya es más difícil buscar algo en común, más aún sin perder mucha información. A menudo este proceso es difícil y supone una gran cantidad de tiempo.

1.3.2 One-hot encoding

, Este método consiste en coger los posibles valores de un atributo, convertirlos en atributos. Supongamos que inicialmente tenemos un atributo "Clase", que tiene tres posibles valores, clase A, clase B y clase C. Este método consiste en eliminar el atributo "Clase", y crear tres atributos "Clase A", "Clase B" y "Clase C". Una observación que inicialmente, para el atributo clase, tenía el valor "Clase A", se marcará con un 1 en el nuevo atributo "Clase A", y con un 0 en los otros dos.

1.3.3 Agrupamiento de datos: Data Binning

Este proceso consiste en perder cierta información a cambio de un formato o valor más conveniente, mediante la transformación de datos continuos en datos discretos o categorías, normalmente perdiendo cierto grado de información. Se entiende con el siguiente ejemplo: supongamos que estudiamos el valor de una casa, y uno de los atributos que tenemos son las medidas de la pista de tenis. En este caso puede ser conveniente perder las medidas concretas de la pista de tenis, y quedarnos únicamente con si tiene pista o no, pues al fin y al cabo es lo que marca la principal diferencia en el precio de una casa, y nos importa menos si una pista tiene 2 metros cuadrados más o menos. Cambiamos las medidas por true/false o por 1/0 (one-hot).

1.3.4 Información que falta

En muchas ocasiones nos encontramos con que tenemos datos imperfectos, como observaciones en las que nos falta el valor de un atributo, por ejemplo. Para "arreglar" la observación, rellenamos el campo que nos falta con un valor artificial, que suele ser la media o la moda obtenida para ese atributo de entre todas las observaciones.

1.3.5 Conjunto de entrenamiento y conjunto de test

Este método supone un gran avance en el aprendizaje automático supervisado. Consiste en utilizar solo un porcentaje de los datos, normalmente entorno a un 80%, para que el modelo aprenda, y luego, cuando llegamos al porcentaje de acierto buscado, realizamos predicciones sobre el conjunto de test. La ventaja de esto es que el conjunto de test nos da una idea de cómo funcionará nuestro modelo en predicciones reales, pero de los que conocemos el resultado. Además, evita el sobre-ajuste que ocurre cuando el modelo intenta aprender con excesiva precisión de los datos de los que disponemos. Esto se debe a que en cualquier muestra real tiene lugar cierto ruido que nunca conviene tener en cuenta.

El conjunto de test supondrá la prueba final que decida si el método ha aprendido correctamente o si hay que volver al proceso de aprendizaje ajustando los hiper-parámetros, es decir, la configuración del modelo, de la que no se puede explicar mucho en este punto porque depende completamente del modelo del que estemos hablando. Por ejemplo en una red neuronal serán los pesos y desviaciones, y en un algoritmo de clustering con vecinos más próximos será el número de vecinos a estudiar.

En el caso de aprendizaje supervisado se le da el conjunto de entrenamiento y luego se prueba con el test. El error se puede medir como la media total de error. Si es muy grande, puede ser porque no hayamos barajado bien las observaciones o porque haya que revisar los parámetros de nuestro modelo, que afectan directamente a los patrones que el modelo encuentra.

1.3.6 Validación cruzada (Cross Validation)

Consiste en realizar el proceso de entrenamiento y test con distintos conjuntos sobre los mismos datos, más o menos veces, y luego combinando los modelos obtenidos en uno solo. Esto se llama k-fold validation technique, pues se repite el proceso k veces, haciendo grupos de información de #(observaciones totales)/k, y utilizando todos los grupos al menos una vez como parte del conjunto de entrenamiento y al menos una vez como parte del conjunto de test. De esta manera se evitan errores como el sobreajuste (overfitting).

2 Regresión

2.1 Regresión lineal

Consiste en aproximar las etiquetas a una línea, es decir, hacer que el valor de la etiqueta dependa linealmente del valor del vector X, minimizando la distancia entre el valor real de la etiqueta y el que le asocia la función lineal que hallemos. Hay una fórmula para hallar esta recta:

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n\sum x^2 - (\sum x)^2}$$

$$b = \frac{n\sum xy - (\sum x)(\sum y)}{n\sum x^2 - (\sum x)^2}$$

2.2 Regresión logística

La regresión logística es un método para resolver problemas de clasificación, en los que las observaciones pueden clasificarse en un tipo u otro (siempre una cantidad discreta de tipos). Para esto, se suele utilizar una función sigmoidea, que nos da una curva con forma de S con la que podemos mandar cualquier dominio al intervalo (0, 1). Cuando hay solo dos posibilidades para el tipo de la observación, se llama regresión logística binomial, y cuando hay más, multinomial. **No entiendo aquí qué tiene que ver la función sigmoidea, no entiendo qué predice si siempre tiene la misma forma.**

3 Análisis de grupos/Agrupamiento/Clustering

Es una técnica de clasificación. Tiene un parámetro K que es la cantidad de grupos que va a haber, y el método que utiliza para encontrarlos es asignar K núcleos, inicialmente coincidiendo con observaciones, y asigna a cada observación el grupo cuyo núcleo esté más cerca. Cuando termina, cambia el núcleo por la media de las observaciones de cada grupo, y repite la asignación de observaciones a grupos, tantas veces como haga falta hasta que no se produzca ningún cambio al modificar el núcleo. Además, esto se va haciendo para distintos valores de K, y para escoger el mejor, lo que se hace es una gráfica, que estará formada por segmentos, y nos quedamos con el valor que forme entre los dos segmentos que separa, el ángulo más agudo, es decir, la esquina más pronunciada. La diferencia con análisis de grupos jerárquico es que en análisis de grupos se busca la cantidad de núcleos indicada por K, mientras que en el jerárquico simplemente separa hasta el final, en grupos más pequeños según bajamos de nivel en el árbol.

El libro habla también de K-vecinos más próximos (K-Nearest Neighbours Clustering) como el algoritmo de análisis de grupos para aprendizaje automático supervisado. Aquí los grupos ya están hechos, y se clasifica cada nueva observación añadiéndose al grupo al que pertenezcan la mayoría de sus vecinos. Hay un parámetro que es la cantidad de vecinos que miramos (siempre los más cercanos desde luego). Es bueno coger una cantidad de vecinos impar, para evitarnos el empate clásico. Desventajas: caro computacionalmente, especialmente si hay varias dimensiones. En este último caso se recomienda unir variables, utilizando por ejemplo Análisis de Componentes Principales (Principal Component Análisis).

4 Sesgo/parcialidad y varianza (Bias and variance)

Un problema clásico en aprendizaje automático es encontrar el punto justo de complejidad para nuestro modelo, de tal manera que tenga cierto error en el conjunto de entrenamiento a cambio de tener el menor error posible en el conjunto de test. Normalmente, si intentamos tener un error muy pequeño en el conjunto de entrenamiento, nuestro modelo se hace complejo, pues intenta en cuenta todas las variaciones, tendremos mayor error en el conjunto de test, puesto que muchas de las variaciones son circunstanciales, y no se repiten como el modelo espera. Una técnica para evitar esto es barajar las observaciones antes de separar entre conjunto de entrenamiento y de test, aunque en general siempre hay que variar los hiper-parámetros del modelo hasta conseguir buenos resultados. Otra forma más de intentar acertar en el punto justo de error en ambos conjuntos es la regularización, que consiste en introducir una penalización que se incrementa proporcionalmente a la complejidad del modelo. Por último, la validación en cruz (cross validation) también ayuda.

5 Neural Networks

Notas sobre un video que estoy viendo sobre el descenso de gradiente, y cómo una red neuronal aprende. El "Hello world!" de las redes neuronales resulta que es el reconocimiento de números escritos a mano. Para esto, se divide la imagen en píxeles, cada uno con un número entre 0 y 1 que representa su valor en la escala de grises, siendo 0 el color blanco y 1 el color negro. Esta es la primera capa de la red neuronal, que puede ser de gran tamaño (en el ejemplo 784 neuronas (píxeles)). Con unos pesos y desviación, se van pasando capa a capa estos valores, modificándose cada vez, y cuando llega a la última capa se compara con el valor esperado, comparando el valor numérico de cada neurona con el que tendría que tener, por ejemplo con el módulo de la diferencia o restando ambos valores al cuadrado. Una vez conocemos esta cantidad de error, nos gustaría hacer ajustes para minimizarlo. Para esto, imaginamos la red neuronal como una función. Puede que muy complicada, pero una función, que tiene como entrada el vector de los pesos que hemos dado a todas las neuronas de la primera capa, y como salida, la diferencia entre el vector de salida y el vector esperado. Es muy importante que esto es una función de los pesos, no del vector X. Queremos, dentro de esta función, movernos hacia donde esté el mínimo, pero sin tener que informarnos excesivamente sobre cómo es la función, porque esto es caro. Podemos calcular la pendiente de la función en el punto que hemos obtenido, y si es positiva, es decir, creciente, disminuir el valor de los pesos, y viceversa. En una función con varios mínimos relativos, el que finalmente obtengamos dependerá en gran medida del vector de pesos inicial que elijamos:

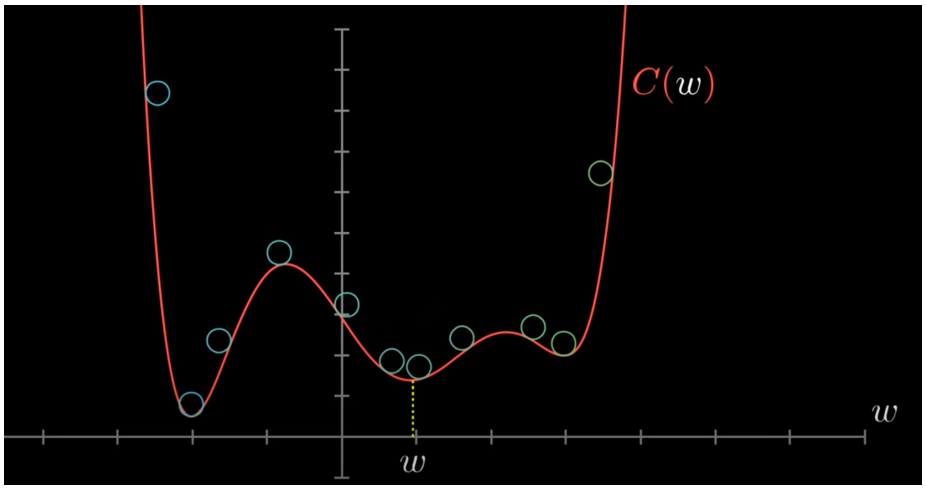

Figure 1: En ésta imagen, los cuatro primeros pesos acabarán en el primer mínimo, los cuatro siguientes en el segundo, y los tres últimos en el tercero.

De esta manera, encontrar el mínimo local es factible, pero encontrar el mínimo global, no. Es una buena idea hacer la variación del peso proporcional a la pendiente. Si hay mucha pendiente podemos suponer que estamos aún lejos del mínimo, y si hay muy poca, que nos estamos acercando. Hasta ahora parece más o menos simple, pero estábamos pensando en una función de una dimensión y una sola imagen, que se representa en un plano. ¿Qué pasa cuando hay más dimensiones, por ejemplo una función de dos variables? Intuitivamente, sabiendo que la imagen de nuestra función es una superficie, seguimos buscando el punto hacia donde rodará una pelota, es decir, queremos la dirección en que descendamos más rápidamente, y aquí es útil saber que el gradiente de una función nos da la dirección de máximo crecimiento, con lo que si nos movemos en dirección contraria (multiplicamos por -1 el gradiente) estaremos "yendo cuesta abajo por la cuesta más empinada posible", que es lo que buscamos.

Este vector gradiente que nos indica en qué dirección mover los pesos, nos dice también qué variaciones de pesos son importantes y cuales no, o al menos, no tanto. La dirección la sabemos mirando el signo de esa componente del vector, y la importancia la vemos en el valor de la coordenada. Si es muy cercano a 0, el cambio no tiene mucha importancia, es decir, esa coordenada ya está bien encaminada hacia el mínimo, o más bien, esté bien o mal, no nos afecta, no nos corre prisa cambiarla. Por ejemplo, si tenemos un vector de pesos (1, 1), y obtenemos un gradiente negativo de (3, 1), sabemos que el cambio en la primera variable es el triple de importante que el cambio en la segunda. Es decir, el cambio en la primera coordenada es más importante que el cambio en la segunda si lo que queremos es descender.

ESTE PARRAFO CREO QUE LO VOY A QUITARAquí hay un matiz importante. Inicialmente estaba imaginando este aprendizaje efectuándose con una sola entrada. Es decir, a partir de un vector de píxeles, se propaga la información hacia la derecha y se llega a un vector de salida. Entonces, yo pensaba que se calculaba el error, y se hallaba el gradiente negativo para propagarlo hacia atrás, de tal manera que los valores del gradiente negativo no tenían más importancia y menos, sino que indicaban en qué dimensiones teníamos que movernos y en cuales no. Es decir, qué valores de qué dimensiones eran prácticamente correctos y cuáles había que modificar. Sin embargo, ahora pienso que se hace la propagación hacia la derecha con muchos vectores, cada uno dando una función de error

6 Párrafos que no sé donde incluir ni si incluirlos o no.

ELIMINAR PÁRRAFO GPU: originalmente surgieron para procesar imágenes muy pesadas muy rápidamente en videojuegos, pero posteriormente se descubrió que se podían usar también para calcular todas las posibilidades en cascada de una red neuronal, obteniendo un tiempo de un día en lo que con una CPU convencional se tardaba varias semanas. La GPU es unidad de procesamiento gráfico, y dice ahí que de lo que estamos hablando es además un "specialized parallel computing chip" un chip de procesamiento especializado en paralelo. Obtiene una ventaja muy grande en el número de operaciones de coma flotante frente a la CPU.

textbfELIMINAR PÁRRAFO¿Qué cantidad de datos necesito? Cuantas más combinaciones de los valores de los atributos tengamos, más sabrá el modelo cómo afecta el valor de cada atributo al resultado o variable dependiente Y. En general, un buen número suele ser 10 veces la cantidad de atributos. Es decir, si tenemos 3 atributos, 30 observaciones.

El valor de la y se llama etiqueta motivación/introducción del problema que quiero solucionar explicar el estado del arte, las técnicas que voy a utilizar y qué cosas hace la gente a día de hoy para solucionarlo. Qué es una red neuronal y tal Qué técnicas voy a utilizar yo. Qué resultados obtengo y eso.

7 Vídeos, artículos y libros de donde he sacado información

https://www.youtube.com/watch?v=lEfrr0Yr684 Aprendizaje automático no supervisado. https://www.youtube.com/watch?v=IHZwWFHWa-w - Gradient descent

8 Notas y dudas

DUDA Estoy traduciendo bias como desviación. DUDA Estoy traduciendo Agent y Environment, de aprendizaje automático reforzado, como agente y entorno. DUDA ¿Es el aprendizaje automático no supervisado siempre para problemas de clasificación? DUDA ¿Es la división de los datos entre conjunto de entrenamiento y conjunto de test propio del aprendizaje automático supervisado únicamente? Librerías y herramientas útiles: sección The ML Toolbox, pag 22, y especialmente a partir de la pag 25.