1. Introducción:

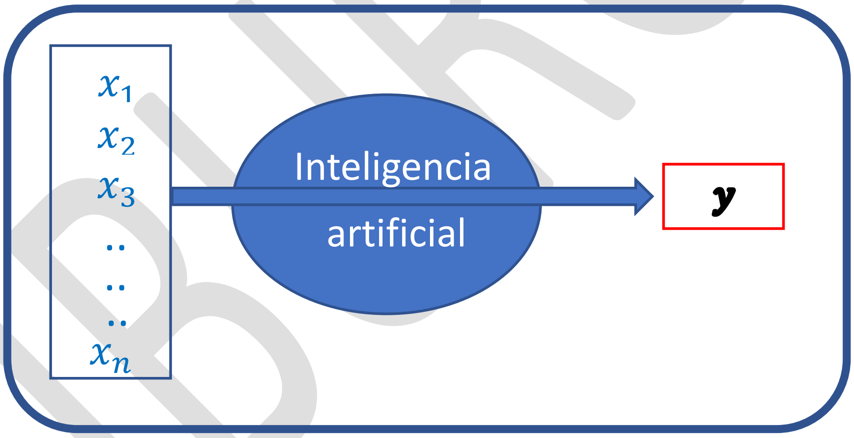
En este trabajo veremos de forma sencilla como una inteligencia artificial decide utilizando varias herramientas matemáticas que normalmente son vistas en los diferentes cursos de bachillerato, pero que nunca se ven con un objetivo de uso real como aplicación. Aquí veremos un uso muy, muy sencillo, pero que nos permitirá entender utilidades matemáticas de:

- La función cuadrática que se ve en varios cursos de secundaria/bachillerato
- La función lineal que se estudia en secundaria en la asignatura de matemáticas
- La recta de regresión lineal y el error que cometemos entre dos variables que se ve en el estudio de estadística de bachillerato, sobre todo en ciencias sociales
- La derivada de una función, vista como aplicación, para estimar otra variable

Es importante que normalmente una inteligencia artificial utiliza muchas variables para determinar otra, y aquí, lo haremos solo en base a una de ellas, pero todos los conceptos anteriores pueden ser generalizados sin ningún problema a 100, 1000 o 100000 dimensiones, es decir, tener en cuenta 100 variables para estimar otra.

2. Input/output para tomar una decisión

Una inteligencia siempre trata de tomar decisiones en base a una serie de datos que almacena, a las variables que representan los datos en los que se basa para tomar la decisión se les llama "**atributos**" y a la variable o característica que tiene que decidir se le llama "**etiqueta**".
Para un matemático, ambas deben considerarse como variables, los atributos serían variables independientes y la etiqueta va a ser una variable dependiente de ellos, así, quedaría:



Lo primero que tenemos que hacer para que una inteligencia artificial pueda tomar una decisión adecuada es **entrenarla** introduciendo tantos datos como dispongamos de situaciones anteriores que hayamos observado, así, la decisión que tomará será más adecuada.

Como ejemplo, podríamos tener como atributos las notas de un alumno en bachillerato, y que la inteligencia artificial le dijese si le considera apto para un grado en concreto, por ejemplo, ADE.

Para entrenar a esta inteligencia deberíamos de tener notas de muchos estudiantes de bachillerato que hayan intentado estudiar ADE, y podríamos decir, que la carrera se ajusta a sus necesidades si es capaz de realizarla en el tiempo estimado de las misma. Otra opción sería calcular el grado de afinidad en base a las notas de bachillerato con varias carreras dadas, en ese caso tendríamos las variables

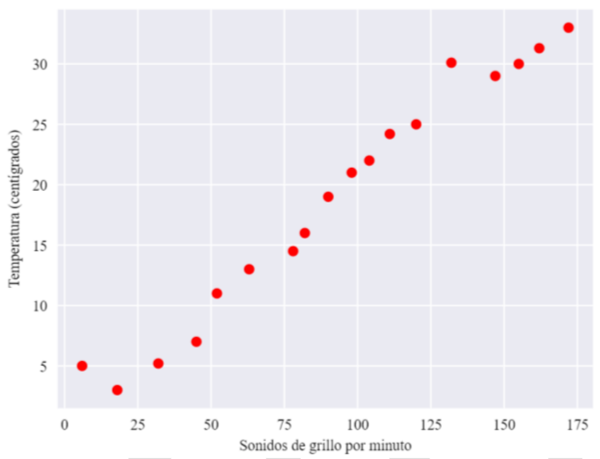
y_1, y_2, y_3, ... Esta recomendación se iría actualizando a medida que se fueran dando recomendaciones, y se fuera entrenando cada vez más la inteligencia artificial.

Pero el caso anterior, tenemos muchas variables de entrada y una o varias de salida que queremos determinar y se escapa del currículo de bachillerato, para que su entendimiento sea entendible por un alumno de bachillerato solo podemos tener una variable de entrada y una de salida.

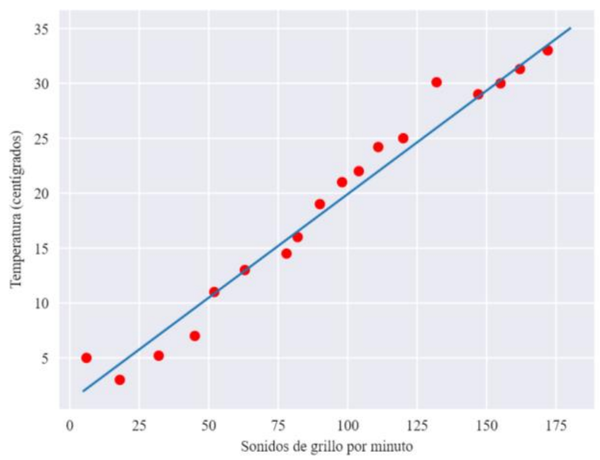
3. Método de regresión lineal

El método de regresión lineal se utiliza cuando se tienen dos variables muy correladas, es decir, que el coeficiente de correlación está cerca de 1 o -1.

Cuando estimamos utilizando el método de regresión lineal representamos todos los datos conocidos de las dos variables en lo que se llama una nube de puntos, donde en el eje de las X representamos nuestro atributo y en el eje de las Y la variable que queremos estimar.



Para posteriormente calcular la recta o función lineal que mejor se ajusta a la nube de puntos, como se ve en el gráfico siguiente.



Así, si tuviésemos que estimar cual es la temperatura de una noche en la que cuento 100 sonidos de grillo por minuto, estimaríamos que hace 20 grados centígrados.

Así calcularemos una predicción de y para un valor de x dado con la fórmula

$$\hat{y} = wx + b$$

Para calcular esta función lineal, es decir, para calcular a y b lo que pretendemos es que el error de equivocarme en la predicción sea el mínimo posible, pero este error lo podríamos calcular de muchas formas posibles, pero en la que nos vamos a fijar es la siguiente. Lo que pretendemos es que el ECM (Error cuadrático medio) sea lo menor posible, que se calcula de la siguiente forma:

$$ECM = \frac{1}{n}\sum_{i=1}^{N}(y_i - \hat{y_i})^2$$

Donde los y_i son los valores que conocemos de esa variable para los datos dados x_i y $\hat{y_i}$ son las predicciones para esos datos, es decir, $\hat{y_i} = wx_i + b$, para unos a y b fijados.

Para calcular estos datos a y b se pueden seguir varios métodos, en general el que se aprende en bachillerato que funciona muy bien cuando tenemos pocos datos es directamente calcular la recta de regresión, cuya fórmula es

$$y - \bar{y} = \frac{\sigma_{xy}}{\sigma_x^2}(x - \bar{x})$$

Donde $\bar{x}$ e $\bar{y}$ son las medias de las variables x e y, σ_{xy} es la covarianza de las dos variables y σ_x^2 la varianza de la variable X, que se pueden calcular también con dos fórmulas matemáticas sencillas, pero no entraremos en ello puesto que para muchos datos no es un procedimiento óptimo de cálculo. Cuando tenemos muchos datos, utilizamos métodos iterativos que con una cantidad mucho menor de cálculos hacemos aproximaciones muy buenas.
A continuación, veremos uno de los más utilizados, aunque no el único, el método de descenso de gradiente

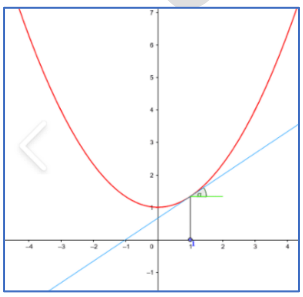
4. Método de descenso del gradiente (en dimensión uno, derivada)

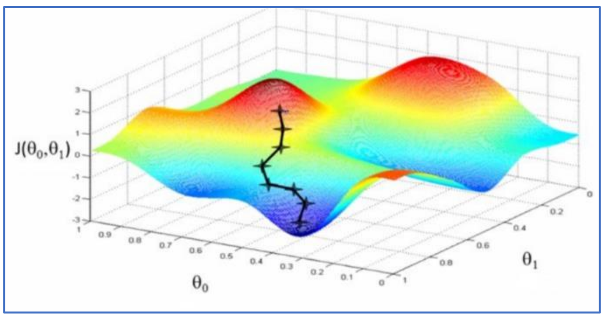
Primero, recordemos que el vector gradiente cuando tenemos una variable, simplemente es la pendiente de la recta tangente en un punto, o sea, la derivada en ese punto x_0, es decir

$f'(x_0) = \frac{df}{dx}(x_0)$

Si tenemos dos atributos, que representamos con las variables x e y a ellos y una etiqueta con la función $z = f(x,y)$, entonces el vector gradiente en el punto (x_0, y_0) es:

$\left(\frac{df}{dx}(x_0, y_0), \frac{df}{dy}(x_0, y_0)\right)$, es decir, el vector cuyas componentes son las derivadas parciales de la función:





Este método consiste en tomar dos valores cualesquiera w y b (una variante es el método de descenso del gradiente estocástico, que comenzaría con dos valores cualesquiera elegidos al azar aleatoriamente), llamado punto de partida.
Vemos que la función “Error cuadrático medio” es una función cuadrática, y, por tanto, tiene pendientes negativas hasta que llegamos a su mínimo y luego positivas a partir de ahí.
Simplemente, se toman dos valores cualesquiera w y b, y se calcula el error cuadrático medio, y si sale negativo, aumentamos la cantidad de ambos en un número a determinar (hay muchas formas de calcular esta cantidad a añadir e implementarlo es el mayor desafío de un programador phyton). Esta cantidad a añadir es conocida como **“Tasa de aprendizaje” o “learning rate”**.

Veamos un ejemplo de una implementación del algoritmo de descenso de gradiente, pero antes veamos cual sería la derivada del error cuadrático medio respecto a la variable w

$$ECM = \frac{1}{n}\sum_{i=1}^{N}(y_i - wx_i - b)^2$$

$$\frac{-2}{n}\sum_{i=1}^{N}(y_i - wx_i - b)$$

Así, que el algoritmo comienza con unos valores para a y b, y va quitar una cantidad α a w y va calculando el valor de la derivada (o gradiente) y si es negativo, suma otra vez, y si es positivo resta,

Es decir, iteramos utilizando $w = w - \alpha\frac{dECM}{dw}$
hasta que la diferencia de errores es prácticamente 0, o sea, menor que 1e-8.

```
trX = np.linspace(-2, 2, 101)
trY = 3 + 2 * trX + np.random.randn(*trX.shape) * 0.33
# Definición de los ajustes y parámetros iniciales
num_steps = 100
learningRate = 0.10
criteria = 1e-8
b_0 = 1
b_1 = 1
# Proceso iterativo
for step in range(0, num_steps):
b_0_gradient = 0
b_1_gradient = 0
N = float(len(trX))
for i in range(0, len(trX)):
b_0_gradient -= (2/N) * (trY[i] - (b_0 + b_1 * trX[i]))
```

```
b_1_gradient -= (2/N) * (trY[i] - (b_0 + b_1 * trX[i])) * trX[i]
b_0 = b_0 - (learningRate * b_0_gradient)
b_1 = b_1 - (learningRate * b_1_gradient)
if max(abs(learningRate * b_0_gradient), abs(learningRate * b_1_gradient)) < criteria:
break
# Impresión de los resultados
print("Los valores que se obtienen son:", b_0, b_1, "en pasos", step)
```

5. Conclusión

En este documento podemos ver la utilización real de varias funcionalidades matemáticas en herramientas que se utilizan en uno de los campos informáticos más utilizados hoy en día, como es la inteligencia artificial.