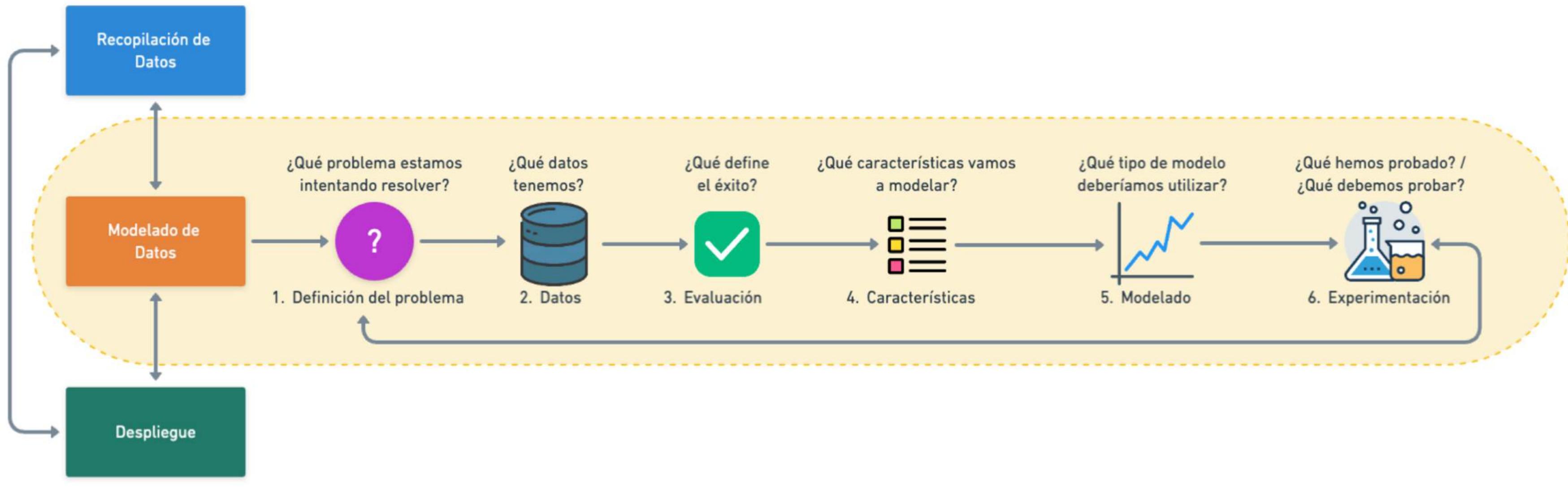


# MACHINE LEARNING

MTRO. ALFONSO GREGORIO RIVERO DUARTE

# FRAMEWORK



# APRENDIZAJE POR REFUERZO

Reinforcement Learning (o aprendizaje por refuerzo) es una rama del Machine Learning, llamada también Online Learning. Se utiliza para resolver problemas de interacción, donde los datos observados hasta el momento  $t$  son considerados para decidir qué acción se lleva a cabo en el momento  $t + 1$ . También es muy utilizada en Inteligencia Artificial cuando se entrena máquinas capaces de hacer tareas como caminar, o conducción de coches inteligentes. Un resultado esperado proporciona una recompensa a la IA, mientras que uno no deseado le otorga un castigo (como si se tratara de educar un humano o un animal). En este caso las Máquinas aprenden a través de ensayo y error.



# TÉCNICAS O ALGORITMOS

# MODELOS DE CLUSTERING

- 1.Upper Confidence Bound (UCB)
- 2.Muestreo Thompson



# **UPPER CONFIDENCE BOUND (UCB)**

# MODELOS DE CLUSTERING

1. El problema del bandido multibrazo
2. Idea de Upper Confidence Bound (UCB)
3. Descripción del problema
4. Algoritmo de Upper Confidence Bound (UCB)



# EL PROBLEMA DEL BANDIDO MULTIBRAZO

# PROBLEMA



Vamos a ver diferentes formas de resolver el problema del bandido del bandido multi brazo y comparar los resultados.

Este problema que vamos a resolver no es el único que se puede resolver con aprendizaje por refuerzo.

Un perro robot es capaz de caminar, es capaz de implementar algoritmos, de caminar, de no chocar con una pared, de subir un escalón o cosas de este estilo.

Y en el caso del robot es súper fácil de entender, porque para enseñarle a caminar hay que enseñarles que primero hay que poner el pie derecho y luego el pie izquierdo de delante, luego el pie derecho de atrás, izquierdo de atrás y en base a ir combinando esos movimientos uno tras otro, es cómo se va enseñando y educando a un robot en forma de perro, a una tarea concreta que es el caminar.

# PROBLEMA

Evidentemente, el algoritmo de aprendizaje por refuerzo lo que hace es entrenar al perro de un modo que básicamente le voy dando las acciones que puede tomar y le enseño cuando mover una pierna es algo positivo y cuando no, cuando da un paso adelante y no se cae, le doy una recompensa. Cada vez que se cae le doy un castigo.

Así que básicamente lo que estoy haciendo es educarlo como si estuviera educando a un animal de verdad o incluso a una persona.

Lo mismo ocurre en el mundo del reinforcement learning. Se le da una recompensa. Normalmente es el número 1 + 1 y el castigo suele ser 0, -1. De modo que lo que quiere el perro es maximizar las recompensas. De ese modo, lo que hace es recordar qué acciones han sido buenas y las intenta repetir cada vez. Y así es como se enseña a caminar a un perro.

# PROBLEMA



¿Qué demonios es un bandido con múltiples brazos? Bueno, lo primero que se nos viene a la mente es un ladrón que va al banco y que tiene toda una banda organizada, múltiples brazos operativos para poder atracar correctamente el banco. Bueno, eso sería la idea más genérica, pero realmente es lo que a nosotros nos preocupa.

Podemos simplificarlo bastante, porque básicamente un bandido multi brazo no es ni más ni menos que una máquina tragamonedas de algún casino.

Antiguamente había que tirar de la palanca para poder arrancar el juego de la máquina. Y el juego que más rápidamente arruina a los jugadores.

# PROBLEMA

Si no recuerdo mal, la probabilidad de que el casino te devuelva el dinero con una máquina tragamonedas no llega siquiera al 50%. Entonces, como es una probabilidad de 50/50, dependiendo de si ganas, no ganas, duplicas, apuesta o pierdes. Pero claro, no siempre 50/50 es ligeramente superior de perder, no de ganar.

A la larga jugando mucho, mucho tiempo, de ahí es donde acabas perdiendo el dinero, y de dónde sale el nombre del bandido. Aquí no hay múltiples brazos, sólo hay uno.

El problema del bandido multi brazo es una especie de juego psicológico cuando entras en una sala de casino, porque en lugar de tener una tragamonedas, hay muchas tragamonedas. Este es el ejemplo completo donde la idea es que cada una de estas máquinas tendrá una programación diferente y llevará más o menos tiempo jugando, es decir, la distribución de premios nos encontrará en el mismo estado.

# PROBLEMA

Se llama multi brazo porque es un problema que sirve para resolver o elegir la mejor tragamonedas que maximiza la victoria del jugador.

Evidentemente, esto tendrá una aplicación directa al mundo del marketing o al mundo de los problemas de una empresa cuando haya elecciones múltiples que queremos elegir una de ellas para maximizar.

El problema es descubrir cuál es la distribución de premios de cada una de estas máquinas. Qué distribución tienen por detrás, una distribución de números, de beneficios, de premios a partir del uso de las mismas y así poder elegir cuál de todas estas máquinas tiene un resultado mejor.

Entonces, la idea inicial sería voy a ir a jugar a ver cuánto gano, cuánto pierdo, cuando hago muchas, muchas, muchas partidas.

# PROBLEMA



D1



D2



D3



D4



D5

# PROBLEMA

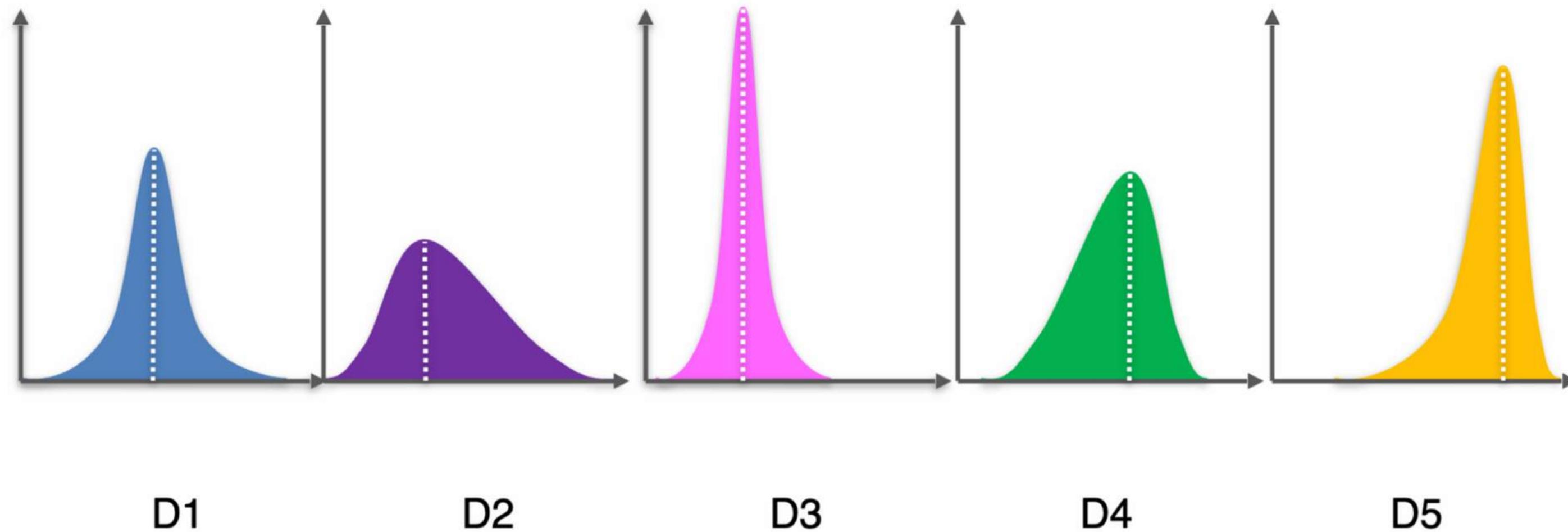
También entendemos que el enfoque de ir y jugar no es el más óptimo, porque acabaré perdiendo todo el dinero antes de descubrir la distribución de la máquina.

Entonces, ¿cuál es la solución para poder enfocar este tipo de problema o que metodología hay que seguir? Pues como de antemano no sé cuál es la distribución de estas máquinas, se me complica un poco.

Aquí tenemos las cinco distribuciones de probabilidad de qué tan probable es que una máquina de uno de los premios cuanto más a la derecha mayor es el premio, y cuanto más alto el pico, mayor es la probabilidad.

¿Cuál es la que da unos premios mayores y por tanto, la que se quiere elegir?

# PROBLEMA



# PROBLEMA

La de la derecha, la distribución número cinco. La de color amarillo.

La mayoría de premios están tirados hacia la derecha. Tiene un sesgo hacia la izquierda. La cola es más larga hacia la izquierda. De modo que los valores más altos, o la media o la distribución concentra unos premios superiores a cualquiera del resto.

Claro, si yo supiera estas distribuciones, evidentemente me iría a la 5.<sup>a</sup> máquina, apostaría el dinero a ella e iría jugando, jugando, jugando.

En promedio, la media de resultados sería superior a la del resto y ganaría más dinero.

Este es el objetivo que quiere resolver reinforcement learning.

# PROBLEMA

Esto tiene una aplicación directa del mundo de la estadística y que se puede resolver de diferentes modos con las técnicas de Reinforcement Learning.

Aquí también entrará en juego un concepto muy divertido que es el del arrepentimiento y el remordimiento, que es un concepto muy divertido porque matemáticamente está definido.

Hay muchísimos matemáticos que han tratado el tema del arrepentimiento y del remordimiento, sobre todo en juegos de azar, y que básicamente todo se trata de buscar los límites de la confianza del jugador, y de eso va precisamente la forma de resolverlo.

El objetivo será encontrar la mejor distribución pasando el menor tiempo jugando y mientras exploras, puedes seguir gastando dinero y ganando y pasando el tiempo. Ese es el objetivo total del ejemplo del bandido multi brazos.

# PROBLEMA

Ejemplo orientado al Marketing:

Imaginemos que tenemos una campaña de publicidad para una gran empresa. Usemos Coca-Cola. La empresa quiere lanzar una campaña que se llama Bienvenido al lado Coke o Al lado Coca-Cola de la vida. Entonces nos propone varios anuncios. Sus artistas gráficos han creado una serie de anuncios, cada uno de los cuales generan datos en base a una campaña de marketing.

Arrancamos cinco campañas de marketing y el objetivo es averiguar qué anuncios funciona mejor para maximizar nuestro retorno. Al inicio, cuando saco la publicidad, no conozco cómo va a funcionar la distribución. Solo la voy a poder conocer después de que miles y miles y miles de personas hayan visto este anuncio y hagan clic o no hagan clic en ello. Esto es lo que me va a dictaminar el éxito de una campaña por encima de las otras.

# PROBLEMA



D1



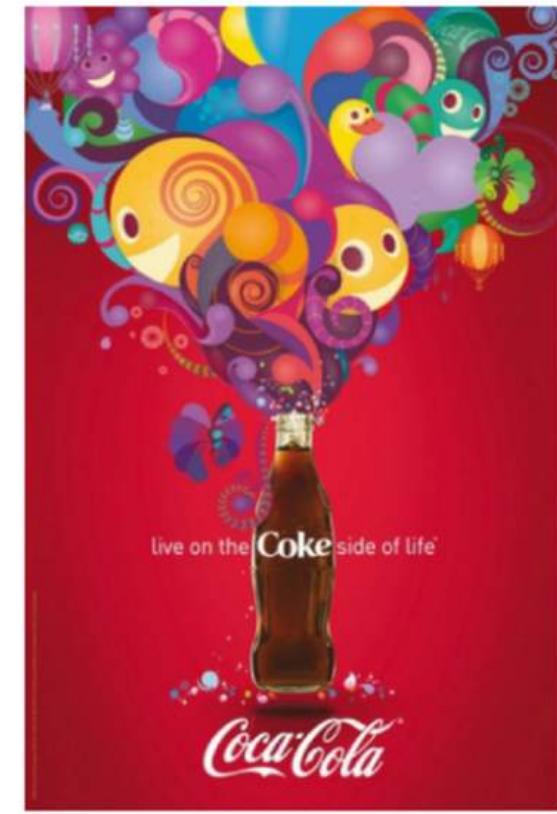
D2



D3



D4



D5

# PROBLEMA

Básicamente se toman los cinco anuncios, o los 50 o los 500 que hay, y se hace un test a gran escala con múltiples pruebas A y B hasta tener una muestra lo suficientemente grande y concluir con cierto nivel de confianza.

¿Qué anuncio otorga mejor resultado?

El problema está en que se suele pasar mucho tiempo y se suele gastar mucho dinero para hacer esto bien.

Lo que podemos hacer es descubrir las distribuciones de cada uno de estos anuncios en lugar de un test A/B,

Pasando desde las distribuciones anteriores y ahora traduciéndolo al éxito de las cinco campañas de Coca-Cola, nos quedaríamos con la número cinco. Evidentemente, el desafío es hacerlo en el menor tiempo posible, intentando sacar el mejor resultado.



# **IDEA DE UPPER CONFIDENCE BOUND (UCB)**

# UPPER CONFIDENCE BOUND (UCB)



El límite de confianza superior, o mejor dicho, upper confidence bound, es un algoritmo que entra dentro de la rama del refuerzo por aprendizaje automático.

Tomando el ejemplo del bandido multibrazo en las máquinas tragamonedas, el problema es saber cómo apostar para maximizar las devoluciones de las máquinas en base a intentar descubrir las distribuciones que existen detrás de las mismas.

Hay una distribución para cada una de ellas y la desconocemos. Por tanto, no sabemos muy bien cuál de todas es la óptima. Por tanto, necesito combinar la exploración de estas máquinas haciendo apuestas para intentar extraer información.

# UPPER CONFIDENCE BOUND (UCB)

La aplicación moderna de este problema es entrar dentro del mundo de la publicidad, de modo que si tenemos 5, 10, 15, 500 anuncios diferentes, hay que averiguar cuál de ellos es el mejor.

Por supuesto, se puede hacer sólo con una prueba de test A/B y luego utilizar los resultados para analizar. Pero eso significa que estamos explorando los resultados, no explotando las distribuciones, descubriendo qué distribución tiene cada anuncio detrás. Eso significa que perderemos mucho tiempo.

¿Cómo se puede afrontar este problema? Los brazos pueden ser los anuncios que mostramos a los usuarios cuando se conectan a una página web o podrían ser cada una de las máquinas. Y el objetivo es elegir qué anuncio tengo que mostrar cada vez o a qué máquina tengo que jugar cada vez.

# UPPER CONFIDENCE BOUND (UCB)

- Tenemos **d** brazos. Por ejemplo, los brazos son anuncios que mostramos a los usuarios cuando se conectan a una página web.
- Cada vez que un usuario se conecta a la página web, se desencadena una ronda.
- En cada ronda, **n**, se elige uno de los anuncios a ser mostrado al usuario.
- A cada ronda **n**, el anuncio **i** da una recompensa:  $r_i(n) \in \{0, 1\}$

$r_i(n) = 1$  Si el usuario hace click en el anuncio i en la ronda n

$r_i(n) = 0$  Si el usuario no hace click en el anuncio i en la ronda n

- Nuestra meta es maximizar la recompensa a través de las rondas que se lleven a cabo.

# UPPER CONFIDENCE BOUND (UCB)

Para ello, fijemos que en cada ronda N se elige uno de los anuncios a ser mostrado, o en su caso, a una de las máquinas a ser jugadas. En el enfoque clásico del problema del bandido multi brazo, sólo puedo tirar de uno de los brazos cada vez. Sólo puedo jugar a una de las tragamonedas cada vez.

En cada ronda que juegue o cada anuncio que muestre, voy a otorgar una recompensa, el anuncio y dará una recompensa en forma de número cero o uno. De acuerdo, si el usuario hace clic en el anuncio, si el usuario tiene un beneficio en la máquina, pues la recompensa para el anuncio y en cada ronda N será uno y cero.

En otro caso, por tanto, cuantificado las recompensas que recibe el usuario en cada momento y como en principio, todo esto será diferente, el objetivo será maximizar la recompensa a través de las rondas que decidamos llevar a cabo.

# UPPER CONFIDENCE BOUND (UCB)

Esto es básicamente el enfoque algorítmico de lo que vamos a llevar a cabo para resolver el problema del bandido multi brazo.

Estos serán los pasos por los que va a pasar nuestro algoritmo y la solución. La receta de cocina de esta clase, la confidence bound se basa en calcular precisamente un intervalo de confianza.

En este caso serían tres pasos que hay que tener en cuenta. No voy a entrar en demasiados detalles de cómo se llega a la fórmula exacta. Lo que quiero es la esencia del algoritmo, no cómo se llega hasta él.

De acuerdo, simplemente a cada ronda se calculará el número de veces que el anuncio se muestra en dicha ronda, la suma de las recompensas hasta dicha ronda, se calcularán la recompensa promedio y la amplitud del intervalo de confianza y elegiremos el anuncio que tenga un límite de confianza superior.

# UPPER CONFIDENCE BOUND (UCB)

**PASO 1:** A cada ronda  $n$ , se consideran dos números para cada  $i$

- $N_i(n)$  El número de veces que el anuncio  $i$  se selecciona hasta la ronda  $n$ ,
- $R_i(n)$  La suma de las recompensas del anuncio  $i$  hasta la ronda  $n$

**PASO 2:** A partir de estos dos números calculamos

- La recompensa media del anuncio  $i$  hasta la ronda  $n$

$$\bar{r}_i(n) = \frac{R_i(n)}{N_i(n)}$$

- El intervalo de confianza en la ronda  $n$ :

$$(\bar{r}_i(n) - \Delta_i(n), \bar{r}_i(n) + \Delta_i(n)) \quad \text{con } \Delta_i(n) = \sqrt{\frac{3 \log(n)}{2N_i(n)}}$$

**PASO 3:** Se selecciona el anuncio  $i$  con mayor límite superior del intervalo de confianza (UCB):

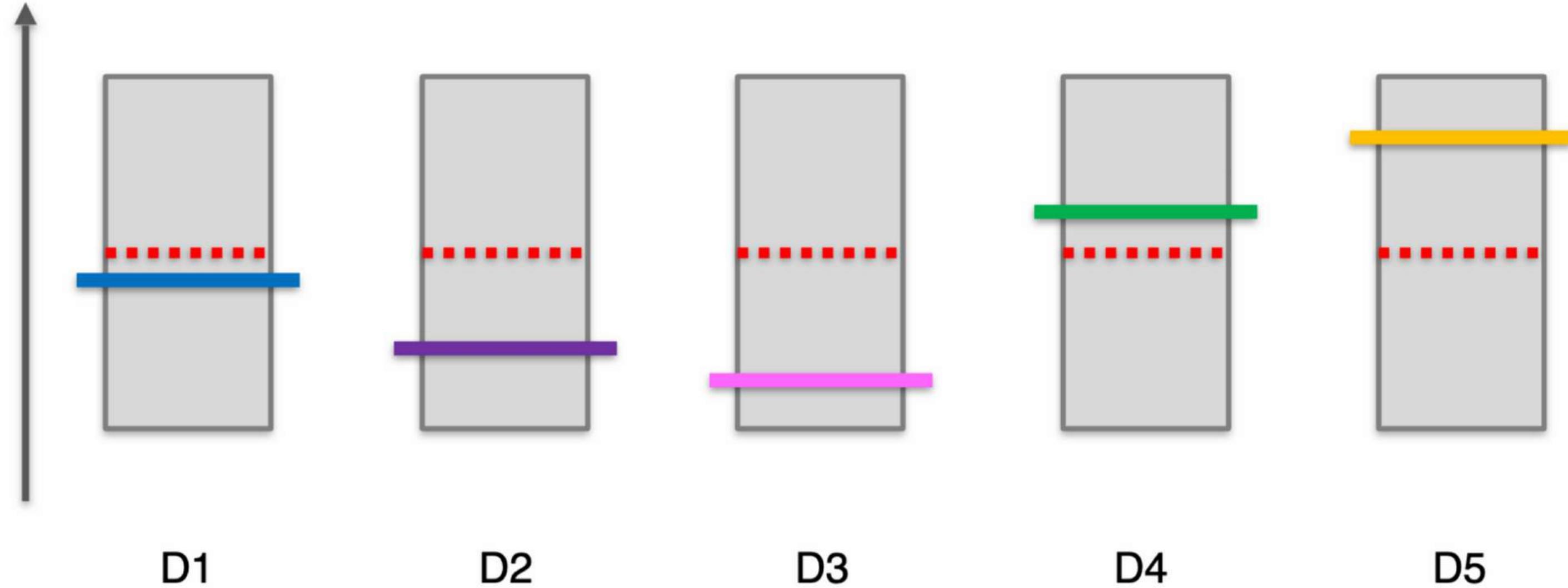
# UPPER CONFIDENCE BOUND (UCB)

Recordemos nuestro problema del bandido multibrazos. Donde hice un poco de trampa y tenemos las 5 distribuciones donde ya se dijo que la 5ta es la más óptima.

Entonces, si yo ya conozco la distribución, deberíamos intentar cuantificar. Que el algoritmo me devuelva esa por encima del resto. Y necesitamos saberlo casi de inmediato en base a ir jugando.

Bueno, vamos a intentar transferir esta información de las distribuciones en lugar de en horizontal en un eje vertical. De modo que vamos a hacer un flip así de la información, y así es como nos va a quedar la distribución. Las rayitas que tenéis en horizontal de cada uno de los colores representan el valor esperado o el valor medio de cada una de las distribuciones.

# UPPER CONFIDENCE BOUND (UCB)



# UPPER CONFIDENCE BOUND (UCB)

Pero de nuevo, yo no conozco el valor exacto, solo el valor medio a base de jugar. Al inicio lo único que puedo hacer es suponer de forma teórica que todas tienen la misma distribución de la línea de puntos. (roja)

Asumimos que todas ellas son iguales hasta que tenga evidencias de demostrar lo contrario, que todas tienen el mismo retorno. Pues el algoritmo lo que hace es, aplicando esas fórmulas, construir la caja que se ve alrededor.

Cada uno de ellos crea una banda alrededor de la misma que representa el nivel de confianza del beneficio de la máquina.

Y aquí lo que estamos intentando buscar es cuál de ellas tiene la caja con el tope superior más arriba. De modo que al inicio todas y cada una de ellas serían exactamente iguales. El nivel de confianza sería el mismo para todos y cada uno de los resultados esperados.

# UPPER CONFIDENCE BOUND (UCB)

Y básicamente, lo que tengo que ir haciendo es rondas de prueba para primero mover las líneas discontinuas de color rojo para probar la media de las máquinas por un lado, y que la banda se vaya ajustando cada vez a la horizontal de fija, a la vez que se va recalculando la anchura del intervalo de confianza y se va moviendo tanto el límite superior como el límite inferior. Esa es la idea general del algoritmo.

Evidentemente, si tuviera dinero infinito y paciencia, sobre todo al final, ¿que pasaría? Las líneas de color rojo discontinuo acabarían exactamente encima de las de colores de la media general de cada distribución.

Y cuantas más pruebas hiciera, menor sería el intervalo de confianza, de modo que el tamaño de la caja iría decreciendo con el tiempo.

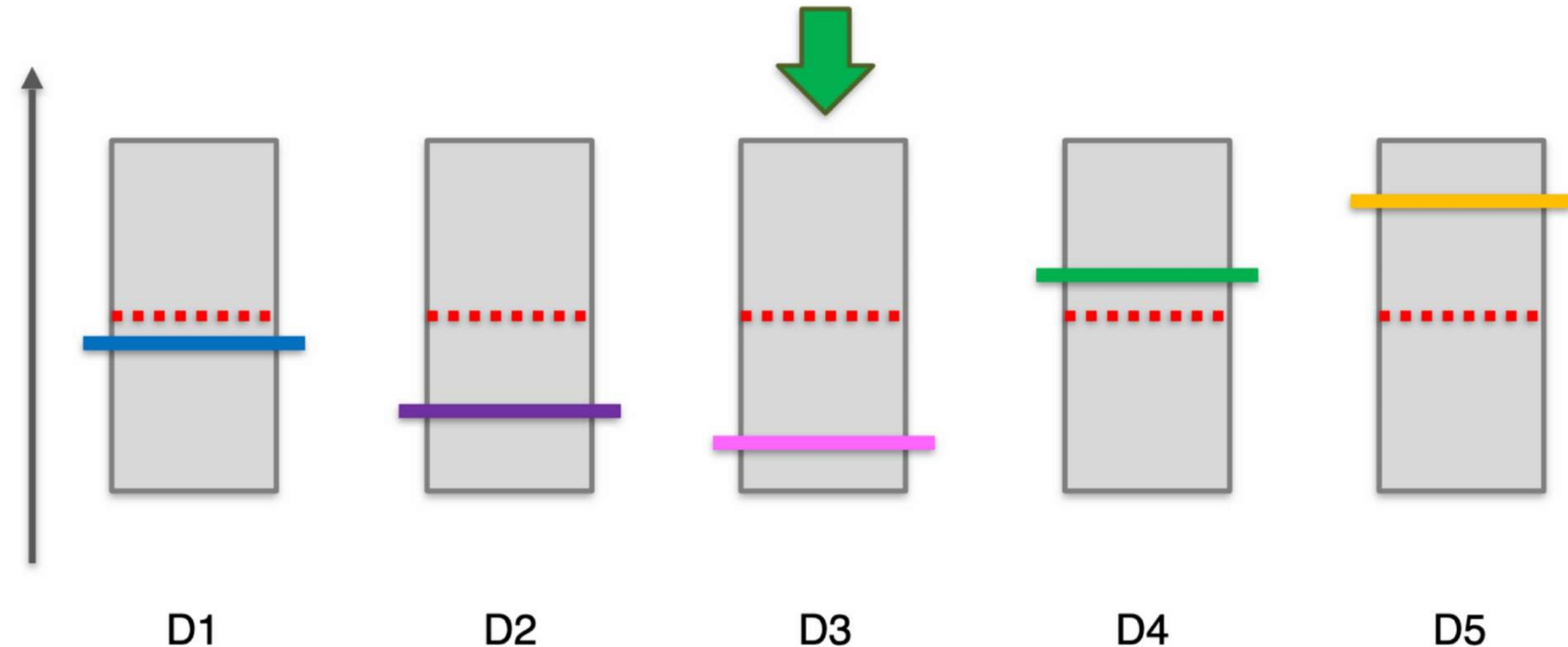
# UPPER CONFIDENCE BOUND (UCB)

¿Qué sucedería si no ejecutara nada? Todos los límites superiores son iguales. No podría decidir. Bueno, pues vamos a hacer una tirada de la tragamonedas. O lo que es lo mismo, muestro un anuncio a una persona y si hace clic en él o no hace clic en él, obtengo un resultado uno o cero respectivamente.

Imaginemos que en este caso la persona no hace clic en este anuncio. La media bajaría ligeramente con respecto a la media general. La media del anuncio o de la tragamonedas, que es lo mismo, ha bajado ligeramente y que el intervalo de confianza se ha quedado por debajo, con el límite por debajo de lo que estaba anteriormente.

Entonces esto es debido a que en esta, en este caso, el anuncio que hemos mostrado o la tragamonedas que hemos usado no ha tenido ningún premio.

# UPPER CONFIDENCE BOUND (UCB)



# UPPER CONFIDENCE BOUND (UCB)

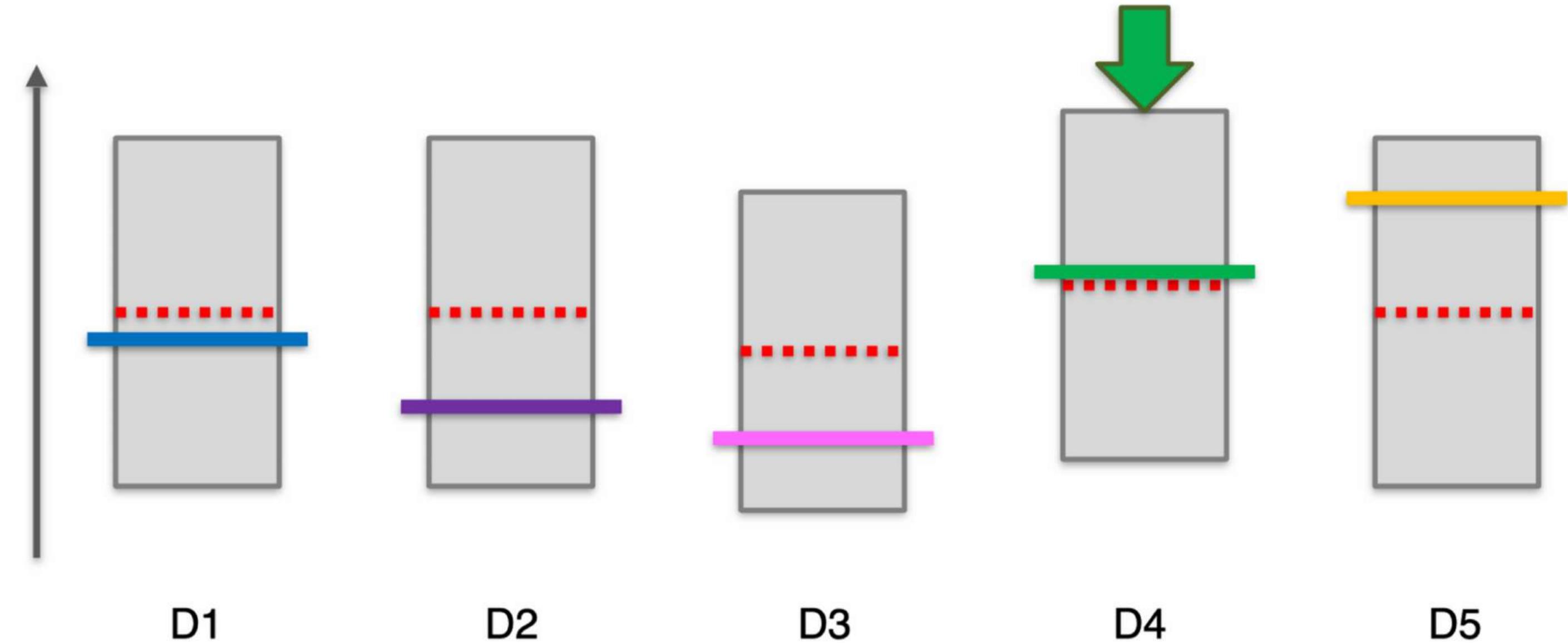
Aquí se agrega toda la información de todo lo observado con respecto a esta máquina. El valor de puntos discontinuos baja porque la media es ligeramente inferior a como era anteriormente y que la cajita de momento sigue siendo del mismo tamaño porque solo tengo una observación.

La ley de los grandes números me ampara y me dice que cuando  $N$  sea bastante grande, la media convertirá la media de la distribución, por tanto es normal que haya bajado.

Esto suele pasar siempre que calculas intervalos de confianza. Cuanto mayor es la  $nN$  menor suele ser la amplitud del intervalo. Esto tiene que ver con el mundo de la estadística inferencial y realmente es un proceso de convergencia muy lento porque el tamaño de la caja va a ir bajando paulatinamente.

El siguiente paso vamos a tener que jugar con otra máquina, eligiendo otra totalmente al azar y repetir el experimento.

# UPPER CONFIDENCE BOUND (UCB)



# UPPER CONFIDENCE BOUND (UCB)

En este caso mostramos el anuncio. La persona puede hacer clic o no hacer clic en él y eso afectará al promedio que hemos medido hasta el momento o el empírico del que habíamos partido, la línea de puntos discontinua.

Entonces, en este caso, la línea empírica cambiará paulatinamente. En este caso ha subido ligeramente la media y vemos que ya empieza a converger hacia el valor real, el valor de la distribución de esa máquina y que por ahora es la que está por encima de todo. Si esto fuera la iteración final, nos quedaríamos con esta máquina, con la distribución número cuatro y comenzaríamos a explotarla.

Por tanto, el algoritmo sería un poquito inútil. Y es que a pesar de haber incrementado la media, hay que ajustar el límite de confianza de ese intervalo y por tanto, hay que reducirlo según la fórmula anterior.

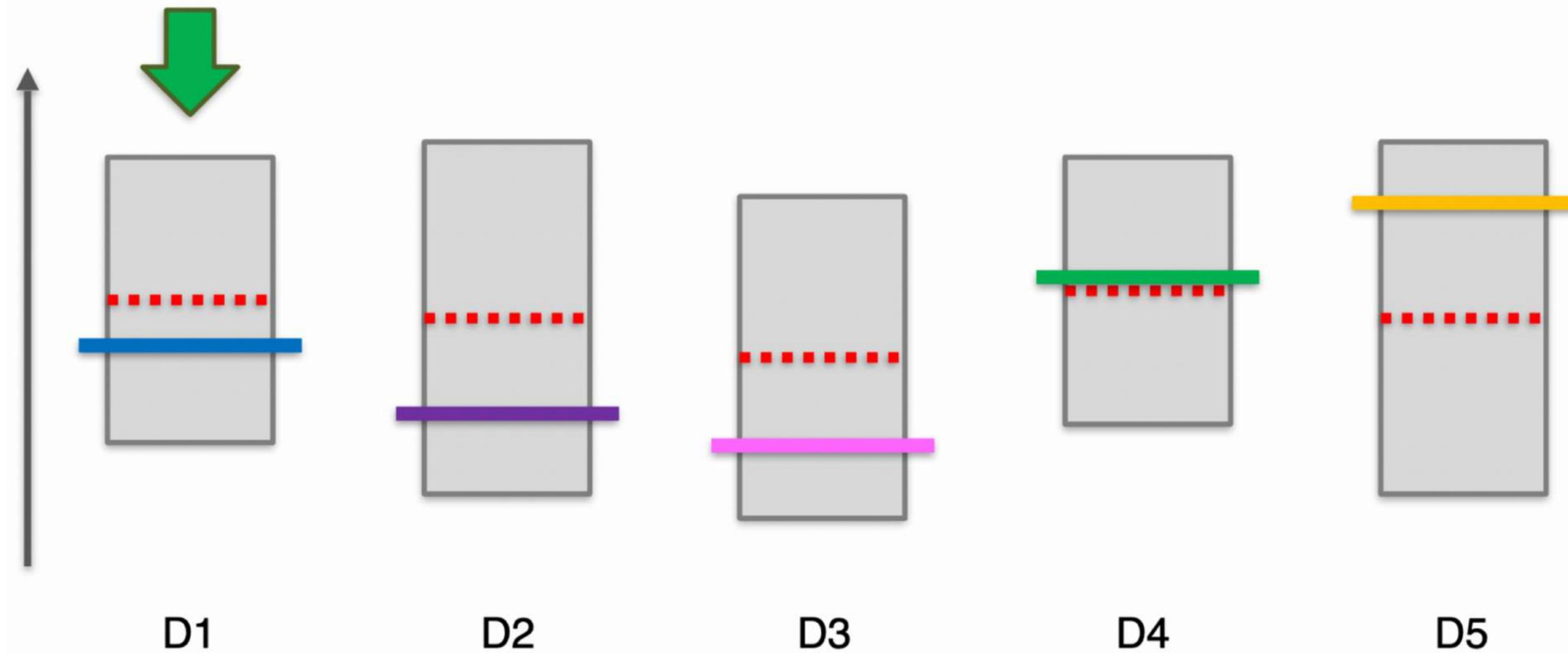
# UPPER CONFIDENCE BOUND (UCB)

Elegimos otra totalmente al azar, y empezamos el algoritmo de nuevo. Vamos a elegir la primera.

Aquí la línea roja ya está bastante cerca de la azul. Evidentemente la teoría nos dice que si ahora tiro la moneda, la roja debería acercarse a la azul. Pero claro, con un solo resultado tampoco se puede garantizar al 100%. Podría ser que justamente se tira la moneda o se muestra el anuncio y se hace clic en él, de modo que ligeramente sube. No se debe pensar que se va a converger con cinco tiradas, hay que hacer unas cuantas cientos de ellas.

Entonces podría pasar que alguien por casualidad, haga clic en ella. Se trata de que a la larga va a converger, pero en una ocasión aleatoria podría subir como acaba de pasarme a mí.

# UPPER CONFIDENCE BOUND (UCB)



# UPPER CONFIDENCE BOUND (UCB)

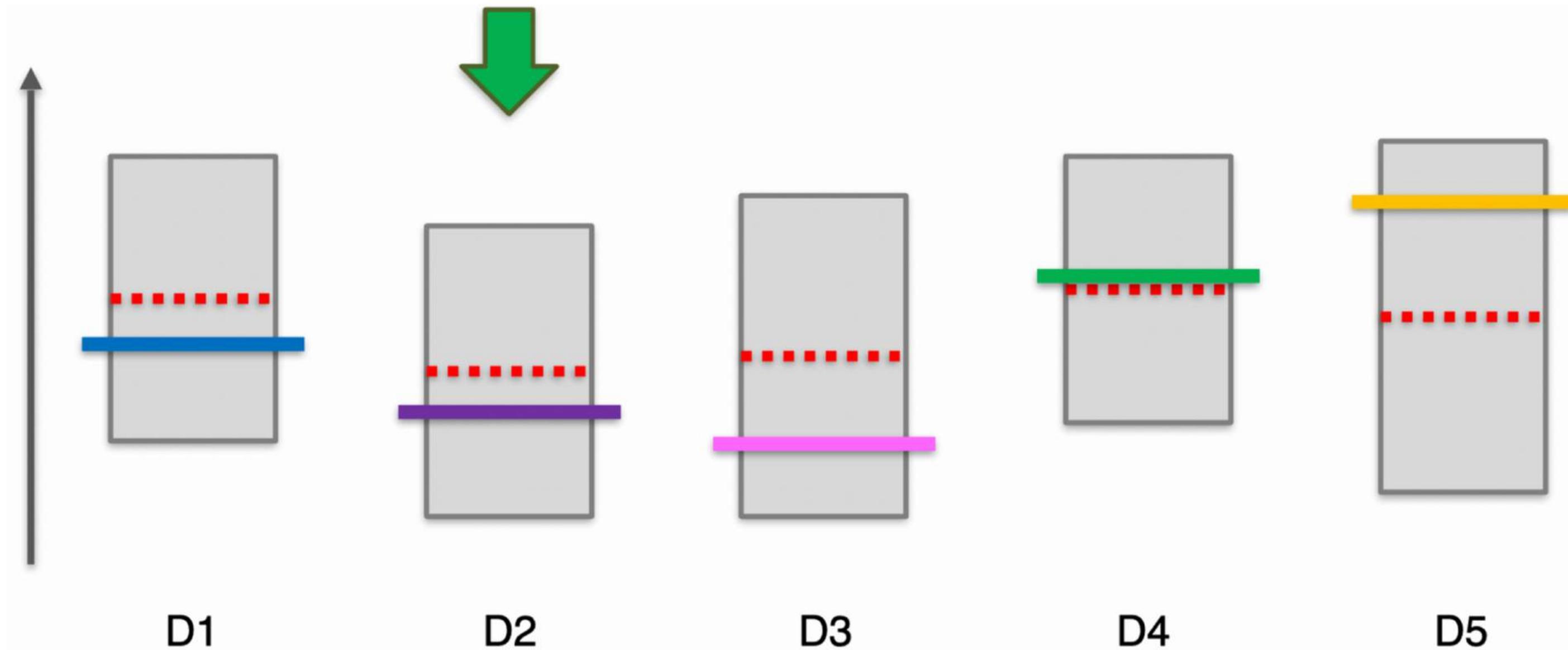
,Podríamos ir ejecutando esto en base a muchas, muchas rondas.

Elegir de nuevo otra máquina aquí podría ganar o no ganar dinero. La máquina tragamonedas o el usuario probablemente no gane. De modo que esto baja la media del intervalo de confianza por un lado, y se acerca más al valor esperado.

De modo que corregimos también la longitud del intervalo de confianza y observamos que por ahora, con lo que llevamos, parece ser que el que tiene mayor intervalo de confianza superior en la confianza es la distribución número cinco.

Pero claro, cuando estamos en el sector de los negocios o de las máquinas, no me puedo basar en cuatro iteraciones, tengo que seguir iterando.

# UPPER CONFIDENCE BOUND (UCB)



# UPPER CONFIDENCE BOUND (UCB)

De modo que ahora me toca la quinta. Y como el resultado final, lo más probable es que la gente haga clic. Sabemos que este es el mejor anuncio o la mejor máquina, pero lo estamos testeando desde el punto de vista de la estadística.

Por tanto, podría ser que la media subiera. De hecho, sería lo más normal de que de un resultado positivo al usuario de que la persona que está utilizando esta máquina obtenga una victoria superior.

Crece el intervalo de confianza, sube la media por un lado y que estrechando porque tengo más elementos en la muestra, obtengo un resultado todavía más concreto.

Podría pararme aquí y ya está, pero lo más normal es cientos de iteraciones, miles de iteraciones hasta poder explotar cual es el mejor resultado. De modo que a lo mejor este seguirá subiendo. Cada vez obtendré mejor resultado.

# UPPER CONFIDENCE BOUND (UCB)

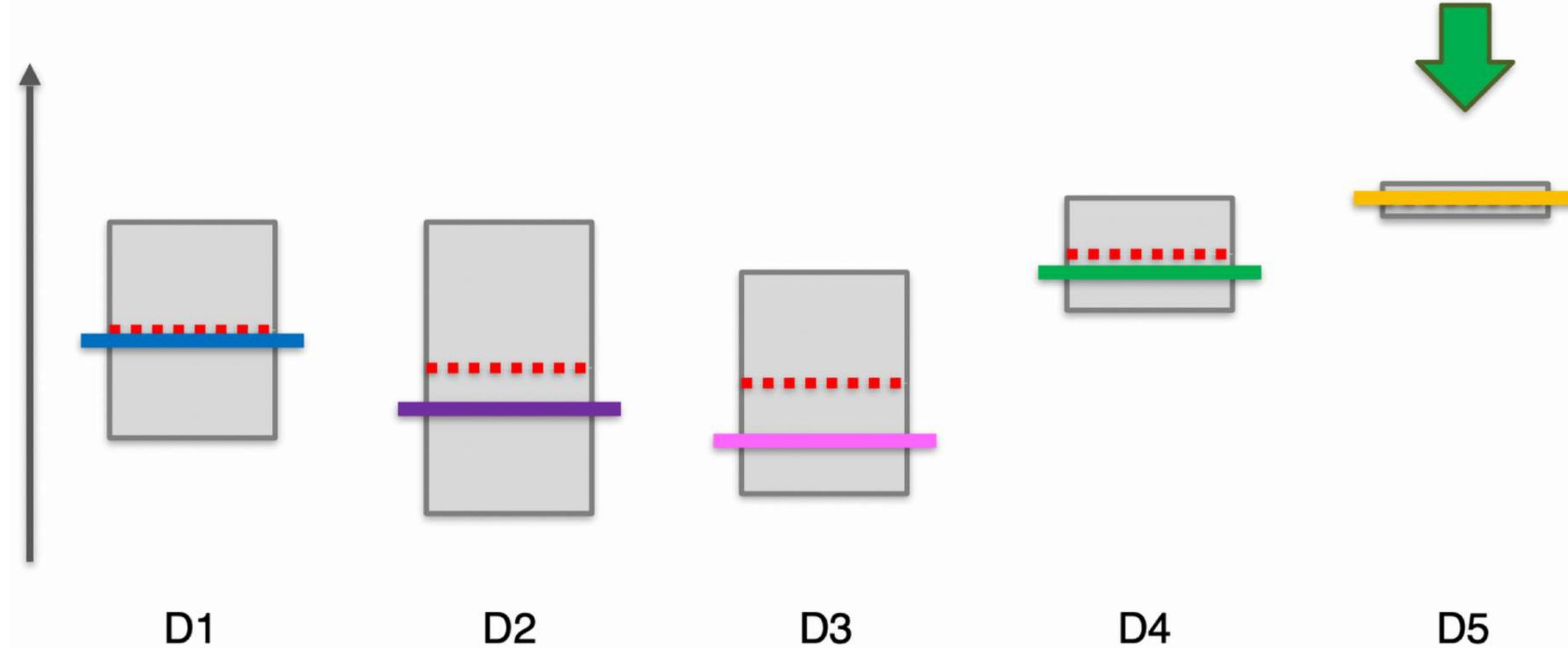
Sigo jugando, sigo jugando, sigo jugando...

El algoritmo va convergiendo hacia un resultado bastante interesante. Ojo, podríamos llegar a un resultado que fuera óptimo, de modo que de vez en cuando el algoritmo tendrá que cambiar a otra máquina para no encontrar un resultado que justamente sea malo.

Podría ser que justamente la de mejor distribución estuviera por debajo, porque una de las veces ha fallado y me estoy centrando en la máquina que no toca. Por tanto, de vez en cuando hay que pasar a otra máquina, como por ejemplo esta de aquí la siguiente en orden de óptima y hacer una tirada para ajustar el intervalo de confianza.

Converge exactamente al valor teórico de la distribución y es un resultado más óptimo que una prueba de test A/B sin más. Porque aquí lo que he ido es a probarlo todo y luego quedarme siempre con el mejor de todos los resultados conocidos hasta el momento.

# UPPER CONFIDENCE BOUND (UCB)





## **DESCRIPCIÓN DEL PROBLEMA**

# UPPER CONFIDENCE BOUND (UCB)

Nos estamos acercando al campo de la inteligencia artificial, porque los robots en general se construyen en base al aprendizaje por refuerzo, en los coches inteligentes, el perro robot y demás.

No vamos a construir ningún robot, sino que lo que vamos a hacer es resolver un problema muy interesante llamado el problema del bandido de múltiples brazos.

# UPPER CONFIDENCE BOUND (UCB)

Tenemos unos datos procedentes del mundo del marketing.

Aquí tenemos un total de diez anuncios de publicidad. Imagina que es la campaña de Coca-Cola, con diez variantes, con diez creatividades. Se basa en clasificar cual de ellas es la mejor y para ello tenemos usuarios, por ejemplo, de una red social, muchas empresas de marketing que utilizan las redes sociales para publicitar Facebook, Instagram, Twitter y demás.

Entonces el departamento de marketing colgó los anuncios en la red social. El departamento de marketing se encarga de preparar diez versiones de ese mismo anuncio que se ponen en la red social. Entonces tienen diez grandes versiones del mismo anuncio y las tienen preparadas, pero no todas ellas se pueden poner en producción como tal.

# UPPER CONFIDENCE BOUND (UCB)

Las diez se ven geniales y realmente no hay nadie en la empresa capaz de decidir una por encima de la otra. Entonces, como no están seguros de qué anuncio poner en la red social, lo que quieren es primero testearlo.

Si quieren colocar el anuncio que obtiene mayor número de clics para evidentemente, intentar maximizar las opciones. Por tanto, buscamos el anuncio con mejor tasa de conversión. Lo que hace la empresa es contratarnos a nosotros como científicos de datos y tenemos diez versiones de un anuncio.

El presupuesto es limitado, pero quiero colocar los anuncios en la red social maximizando la tasa de conversión.

# UPPER CONFIDENCE BOUND (UCB)

Entonces nos contrata a nosotros y nos pide cuál es la mejor estrategia para descubrir rápidamente qué versión del anuncio funciona mejor para no estar tirando el dinero con pruebas y pruebas y pruebas han decidido que el test A/B no es suficiente.

Fijaos que en este caso tenemos información muy diferente a lo que viene siendo el resto del curso. Cuando hablábamos de class training teníamos la variable de las variables independientes, la matriz de características y a partir de ahí teníamos la variable independiente que queríamos predecir la categoría, el valor de regresión y demás.

En este caso, al igual que cuando lo hice con clase, precisamente no sabía muy bien dónde iba a catalogar el resultado, cuál iba a ser el objetivo ganador. Aquí lo mismo. No tengo ni idea de cuál va a salir el resultado.

# UPPER CONFIDENCE BOUND (UCB)

Ni siquiera tengo un proceso de clasificación. Por tanto, todo lo que tengo son variables independientes. Todos son anuncios o resultados de anuncios en la red social.

Y básicamente lo que necesitamos es traducir esto a variables que a un algoritmo sea capaz de trabajar para ello.

Fijaos que tenemos las diez versiones del mismo anuncio. Son las diez columnas propiamente de nuestro data frame. Y básicamente lo que hago es cada vez que un usuario inicia su sesión en la red social en cuestión, le muestro una versión de uno de estos diez anuncios que estará disponible cada vez que el usuario se conecte, de modo que le mostraré una versión. Por ejemplo, al usuario número cero le puedo mostrar el anuncio número tres y voy a observar su respuesta.

# UPPER CONFIDENCE BOUND (UCB)

Si el usuario hace clic en el anuncio, voy a otorgar una recompensa igual a uno. Y si el usuario no hace clic en el anuncio, voy a otorgar una recompensa igual a cero. Por ejemplo, el usuario número cero no hizo clic en el anuncio número tres, y esto lo voy a hacer para 10.000 usuarios de esta red social.

Vamos a mostrar versiones de los anuncios en principio al azar. Lo que pasa es que si lo hacemos al azar, podría ser que un anuncio fuera más mostrado que otro, o que un público viera más que otro y no obtuviéramos el resultado correcto.

Entonces habrá una estrategia específicamente diseñada para que en cada ronda de los resultados previos que hemos observado, influyan en qué vamos a mostrar en el siguiente.

# UPPER CONFIDENCE BOUND (UCB)

Y la clave para entender acerca del aprendizaje por refuerzo es precisamente que tenemos unos diez anuncios. Lo que ocurre es que el algoritmo observará paulatinamente los resultados obtenidos durante las primeras rondas, durante las primeras diez rondas, las primeras diez veces que se muestre el anuncio.

Y en base a esto decidiremos qué anuncio es el que se volverá a mostrar al usuario. De acuerdo, eso forma parte del aprendizaje por refuerzo.

Entonces, como es una estrategia dinámica, depende de las observaciones con las que empiece el algoritmo. Podría ser que un anuncio se mostrara antes que el otro o obtuviéramos resultados incorrectos. Aquí lo que tenemos es una simulación.



# ALGORITMO

# RETO

## Reto:

Comencemos con:

- Active su ambiente local creado con miniconda en su equipo o su código de COLAB
- Abra el archivo [09\\_Algoritmo\\_Apriori.ipynb](#)
- Ejecute el código y analícelo junto al instructor

## Resultado Esperado:

	Left Hand Side	Right Hand Side	Support	Confidence	Lift
3	fromage blanc	honey	0.003333	0.245098	5.164271
0	light cream	chicken	0.004533	0.290598	4.843951
2	pasta	escalope	0.005866	0.372881	4.700812
8	pasta	shrimp	0.005066	0.322034	4.506672
7	whole wheat pasta	olive oil	0.007999	0.271493	4.122410
5	tomato sauce	ground beef	0.005333	0.377358	3.840659
1	mushroom cream sauce	escalope	0.005733	0.300699	3.790833
4	herb & pepper	ground beef	0.015998	0.323450	3.291994
6	light cream	olive oil	0.003200	0.205128	3.114710

# PREGUNTAS Y RESPUESTAS

Mtro. Alfonso Gregorio Rivero Duarte

Senior Data Manager - CBRE

(+52) 5528997069

devil861109@gmail.com

<https://www.linkedin.com/in/alfonso-gregorio-rivero-duarte-139a9225/>

