**Introducción a la Optimización de Hiperparámetros en Azure Machine Learning**

La optimización de hiperparámetros es una parte crucial del desarrollo de modelos de aprendizaje automático. En este módulo, aprenderás sobre cómo ajustar hiperparámetros para mejorar el rendimiento de tus modelos utilizando Azure Machine Learning. Aquí te explico los conceptos clave y los objetivos de aprendizaje que se cubrirán.

**Conceptos Clave**

* **Hiperparámetros**: Son valores que configuran el comportamiento del algoritmo de entrenamiento y no son derivados directamente de los datos de entrenamiento. Ejemplos incluyen la tasa de regularización en la regresión logística y la tasa de aprendizaje en redes neuronales convolucionales (CNN).
* **Tuning de Hiperparámetros**: Este proceso implica entrenar múltiples modelos utilizando la misma combinación de datos de entrenamiento pero variando los hiperparámetros. La evaluación de cada modelo permite identificar cuál de ellos tiene el mejor rendimiento basado en métricas definidas, como la precisión.

**Optimización de Hiperparámetros en Azure Machine Learning**

En Azure Machine Learning, puedes optimizar hiperparámetros mediante el uso de **sweep jobs** (trabajos de barrido). Un sweep job ejecutará pruebas para cada combinación de hiperparámetros especificados, utilizando un script de entrenamiento que logueará el rendimiento alcanzado.

**Objetivos de Aprendizaje**

En este módulo, aprenderás a:

1. **Definir un Espacio de Búsqueda de Hiperparámetros**: Aprenderás a establecer los hiperparámetros que deseas ajustar y el rango de valores que consideras.
2. **Configurar el Muestreo de Hiperparámetros**: Conocerás las diferentes estrategias para seleccionar las combinaciones de hiperparámetros a probar durante el entrenamiento.
3. **Seleccionar una Política de Terminación Temprana**: Esto implica establecer criterios para detener pruebas que no están rindiendo como se esperaba, lo que puede ahorrar tiempo y recursos.
4. **Ejecutar un Sweep Job**: Aprenderás a implementar un sweep job en Azure Machine Learning para optimizar automáticamente los hiperparámetros.

-- **Definir un espacio de búsqueda**

En el contexto del ajuste de hiperparámetros en aprendizaje automático, el **espacio de búsqueda** se refiere al conjunto de valores de hiperparámetros que se probarán durante el proceso de optimización. La definición de este espacio depende del tipo de hiperparámetro que se esté considerando.

**Hiperparámetros discretos**  
Algunos hiperparámetros requieren valores discretos, lo que significa que deben seleccionarse de un conjunto finito de posibilidades. Para definir un espacio de búsqueda para un parámetro discreto, se puede utilizar la clase Choice, la cual permite especificar una lista de valores explícitos, un rango de valores o un conjunto arbitrario de valores separados por comas. Ejemplos incluyen:

* Choice(values=[10, 20, 30])
* Choice(values=range(1, 10))
* Choice(values=(30, 50, 100))

También se pueden elegir valores discretos de varias distribuciones discretas como:

* **QUniform(min\_value, max\_value, q)**: Devuelve un valor redondeado a un múltiplo de q dentro del rango especificado.
* **QLogUniform(min\_value, max\_value, q)**: Similar a QUniform, pero en una escala logarítmica.
* **QNormal(mu, sigma, q)**: Devuelve un valor normalmente distribuido, redondeado a un múltiplo de q.
* **QLogNormal(mu, sigma, q)**: Valor en una escala logarítmica, redondeado a un múltiplo de q.

**Hiperparámetros continuos**  
Los hiperparámetros continuos permiten cualquier valor en un rango, resultando en infinitas posibilidades. Para estos, se pueden utilizar distribuciones como:

* **Uniform(min\_value, max\_value)**: Valor distribuido uniformemente entre los límites especificados.
* **LogUniform(min\_value, max\_value)**: Valor obtenido según una distribución exponencial.
* **Normal(mu, sigma)**: Valor normalmente distribuido con una media y desviación estándar especificadas.
* **LogNormal(mu, sigma)**: Valor en una escala logarítmica, siguiendo una distribución normal.

Captura de pantalla de un celular

Descripción generada automáticamente**Definición del espacio de búsqueda**  
Para definir un espacio de búsqueda, se debe crear un diccionario que contenga la expresión de parámetro adecuada para cada hiperparámetro nombrado. Por ejemplo, el siguiente espacio de búsqueda indica que el hiperparámetro batch\_size puede tomar los valores 16, 32 o 64, mientras que learning\_rate puede tener cualquier valor de una distribución normal con una media de 10 y una desviación estándar de 3:

**Configurar un método de muestreo**

En el ajuste de hiperparámetros mediante un trabajo de optimización (sweep job), los valores específicos utilizados dependen del método de muestreo elegido. Azure Machine Learning ofrece tres métodos de muestreo principales:

1. **Muestreo en cuadrícula (Grid Sampling)**: Intenta todas las combinaciones posibles.
2. **Muestreo aleatorio (Random Sampling)**: Selecciona aleatoriamente valores del espacio de búsqueda.
3. **Sobol**: Agrega una semilla al muestreo aleatorio para hacer los resultados reproducibles.
4. **Muestreo bayesiano (Bayesian Sampling)**: Selecciona nuevos valores basados en los resultados anteriores.

**Muestreo en cuadrícula (Grid Sampling)**

Texto

Descripción generada automáticamenteEl muestreo en cuadrícula se aplica únicamente cuando todos los hiperparámetros son discretos y se utiliza para probar cada combinación posible de parámetros en el espacio de búsqueda. Por ejemplo:

**Muestreo aleatorio (Random Sampling)**

Texto

Descripción generada automáticamenteEl muestreo aleatorio selecciona un valor de cada hiperparámetro de manera aleatoria, lo que permite mezclar valores discretos y continuos. Por ejemplo:

**Sobol**

Texto

Descripción generada automáticamenteEl método Sobol es una variación del muestreo aleatorio que permite reproducir un trabajo de muestreo aleatorio al usar una semilla. Esto ayuda a que la distribución del espacio de búsqueda sea más uniforme. Un ejemplo de uso de Sobol es:

**Muestreo bayesiano (Bayesian Sampling)**

Texto

Descripción generada automáticamenteEl muestreo bayesiano utiliza un algoritmo de optimización bayesiana para elegir los valores de los hiperparámetros. Este método intenta seleccionar combinaciones de parámetros que resulten en un rendimiento mejorado en comparación con selecciones anteriores. Ejemplo de configuración:

Es importante tener en cuenta que el muestreo bayesiano solo se puede utilizar con expresiones de parámetros de tipo Choice, Uniform y QUniform.

-- **Configurar la terminación anticipada**

El ajuste de hiperparámetros es crucial para optimizar el rendimiento de tu modelo y seleccionar los valores que lo harán funcionar mejor. Sin embargo, la búsqueda del modelo óptimo puede volverse interminable, y es importante considerar si vale la pena invertir tiempo y recursos en probar nuevos valores de hiperparámetros.

Cada prueba en un trabajo de optimización (sweep job) entrena un nuevo modelo con una combinación diferente de hiperparámetros. Si el nuevo modelo no ofrece mejoras significativas, puede ser conveniente detener el trabajo de optimización y utilizar el modelo que haya mostrado el mejor rendimiento hasta ese momento.

**Cuándo usar una política de terminación anticipada**

La decisión de implementar una política de terminación anticipada puede depender del espacio de búsqueda y del método de muestreo que estés utilizando. Por ejemplo, si usas un método de muestreo en cuadrícula con un espacio de búsqueda discreto que resulta en un máximo de seis pruebas, podría no ser necesaria una política de terminación anticipada.

Sin embargo, esta política es especialmente útil cuando se trabaja con hiperparámetros continuos en el espacio de búsqueda, ya que estos presentan un número ilimitado de valores posibles. Es recomendable usar una política de terminación anticipada cuando se emplean hiperparámetros continuos y un método de muestreo aleatorio o bayesiano.

**Configuración de una política de terminación anticipada**

Al elegir usar una política de terminación anticipada, hay dos parámetros principales a considerar:

* **evaluation\_interval**: Especifica el intervalo en el que deseas que se evalúe la política. Cada vez que se registra la métrica principal para una prueba cuenta como un intervalo.
* **delay\_evaluation**: Define cuándo comenzar a evaluar la política, permitiendo que se completen al menos un número mínimo de pruebas antes de que la política afecte a los resultados.

Nuevos modelos pueden seguir mostrando solo mejoras marginales respecto a modelos anteriores. Para determinar en qué medida un modelo debe mejorar con respecto a las pruebas previas, hay tres opciones de terminación anticipada:

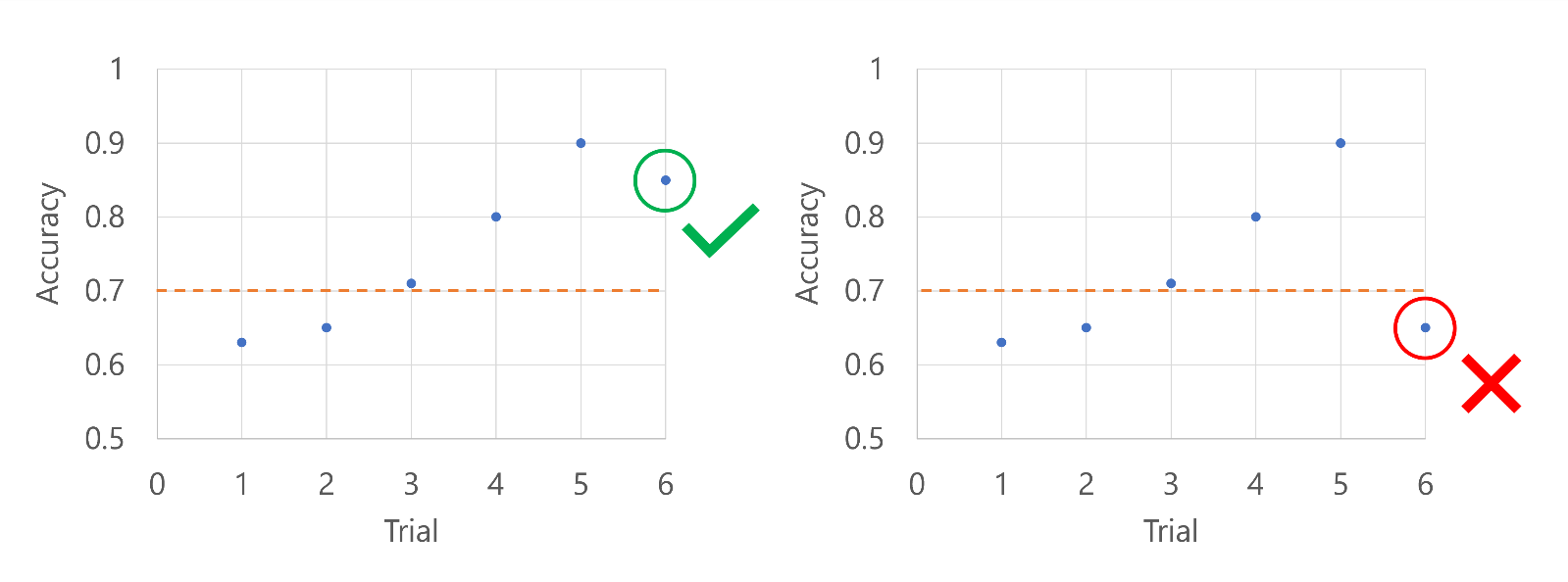
1. **Política de bandido (Bandit policy)**: Utiliza un slack\_factor (relativo) o slack\_amount (absoluto). Cualquier nuevo modelo debe desempeñarse dentro del rango de holgura del mejor modelo hasta el momento.
2. **Política de parada mediana (Median stopping policy)**: Utiliza la mediana de los promedios de la métrica principal. Cualquier nuevo modelo debe superar la mediana.
3. **Política de selección truncada (Truncation selection policy)**: Utiliza un truncation\_percentage, que representa el porcentaje de pruebas de menor rendimiento. Cualquier nuevo modelo debe superar las pruebas de menor rendimiento.

**Ejemplo de Política de Bandido**

Texto

Descripción generada automáticamentePuedes usar una política de bandido para detener una prueba si la métrica de rendimiento objetivo no supera un margen especificado en comparación con la mejor prueba hasta el momento. Aquí tienes un ejemplo de cómo implementar una política de bandido:

En este caso, supongamos que la métrica principal es la precisión del modelo. Después de las primeras cinco pruebas, si el mejor modelo tiene una precisión de 0.9, cualquier nuevo modelo debe tener una precisión superior a (0.9 - 0.2) o 0.7. Si la precisión del nuevo modelo es superior a 0.7, el trabajo de optimización continuará. Si es inferior, la política terminará el trabajo de optimización.



**Aplicar una política de bandido con un factor de holgura**

Además de usar un slack\_amount absoluto, también puedes aplicar una política de bandido utilizando un slack\_factor, que compara la métrica de rendimiento como una proporción en lugar de un valor absoluto. Esto permite evaluar el rendimiento de un nuevo modelo en relación con el mejor modelo encontrado hasta el momento, considerando el desempeño relativo en lugar de una diferencia fija.

**Política de parada mediana**

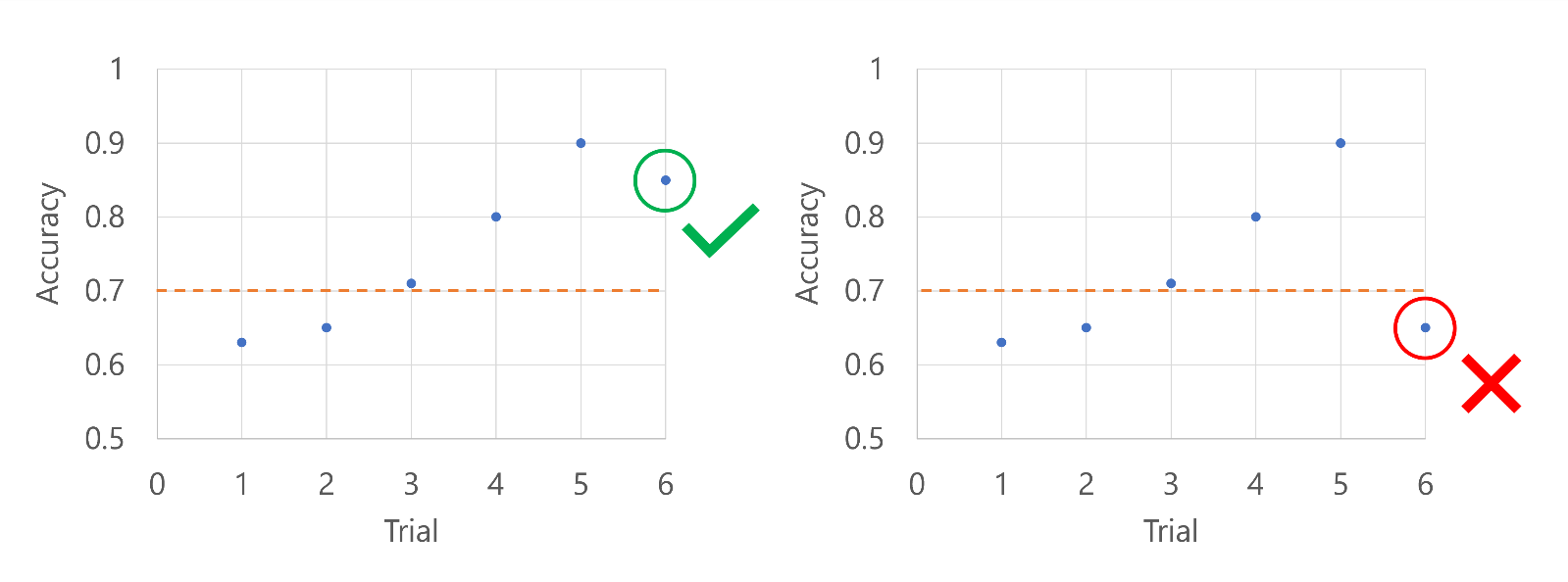
La política de parada mediana (Median Stopping Policy) abandona aquellas pruebas cuya métrica de rendimiento objetivo es inferior a la mediana de los promedios acumulados de todas las pruebas realizadas. Esta estrategia ayuda a centrar la búsqueda en los modelos que tienen un rendimiento mejor que la mediana del grupo.

**Ejemplo de implementación**

Texto

Descripción generada automáticamenteAquí hay un ejemplo de cómo aplicar una política de parada mediana:

En este contexto, supongamos que la métrica principal es la precisión del modelo. Cuando se registra la precisión para la sexta prueba, esta debe ser superior a la mediana de las puntuaciones de precisión acumuladas hasta ese momento. Por ejemplo, si la mediana de las puntuaciones de precisión hasta ahora es 0.82, el nuevo modelo debe tener una precisión mayor a 0.82 para que el trabajo de optimización continúe. Si el nuevo modelo tiene una puntuación de precisión inferior a 0.82, se detendrá el trabajo de optimización y no se entrenarán más modelos.



Además de utilizar un slack\_amount absoluto, puedes aplicar una política de bandido usando un slack\_factor. Esta metodología permite comparar la métrica de rendimiento en forma de proporción en lugar de un valor absoluto, lo que facilita la evaluación del rendimiento de un nuevo modelo en relación con el mejor modelo encontrado hasta el momento.

**Política de Parada Mediana**

La **política de parada mediana** es una estrategia que abandona las pruebas (trials) donde la métrica de rendimiento objetivo es inferior a la mediana de los promedios en curso de todas las pruebas. Esta técnica es útil para optimizar el proceso de selección de modelos, ya que evita la formación de modelos que no superen un umbral de rendimiento considerado adecuado.

**Cómo Funciona la Política de Parada Mediana**

1. **Evaluación de la Mediana**: Durante el proceso de entrenamiento, cada vez que se registra la métrica de rendimiento para un nuevo modelo, se calcula la mediana de las métricas de rendimiento de todos los modelos entrenados hasta ese momento.
2. **Criterio de Continuación**: Si la métrica del nuevo modelo es mayor que la mediana calculada, el trabajo de optimización continuará. Si es menor, se detendrá la búsqueda y no se entrenarán nuevos modelos.

**Ejemplo de Implementación**

Aquí tienes un ejemplo de cómo implementar una política de parada mediana en un trabajo de optimización:

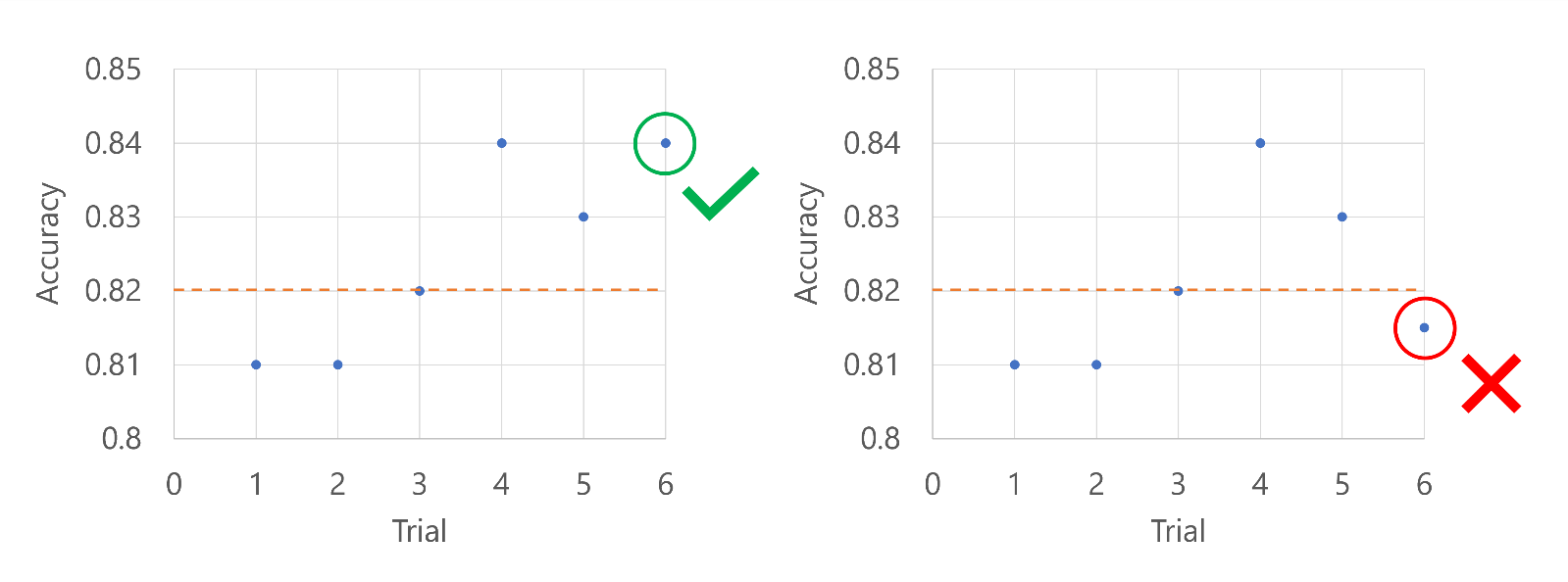
Texto

Descripción generada automáticamente

**Ejemplo de Evaluación**

Imagina que la métrica principal es la precisión del modelo. Cuando se registra la precisión en el sexto trial, la métrica debe ser superior a la mediana de las puntuaciones de precisión anteriores. Supón que la mediana de las puntuaciones hasta ese momento es 0.82:

* Si el nuevo modelo obtiene una precisión superior a 0.82, la búsqueda de hiperparámetros continuará.
* Si el nuevo modelo tiene una precisión inferior a 0.82, la política detendrá la búsqueda de modelos, evitando así el entrenamiento innecesario de modelos que no mejoran el rendimiento.



**Política de Selección por Truncamiento**

La **política de selección por truncamiento** es una estrategia utilizada en la optimización de hiperparámetros que cancela los X% de las pruebas de menor rendimiento en cada intervalo de evaluación, basándose en el porcentaje de truncamiento que especifiques. Esto ayuda a centrar los recursos en las pruebas que tienen un mejor desempeño, mejorando la eficiencia del proceso de entrenamiento.

**Cómo Funciona la Política de Selección por Truncamiento**

1. **Evaluación de Pruebas**: En cada intervalo de evaluación, se analiza el rendimiento de todas las pruebas realizadas hasta ese momento.
2. **Cancelación de Pruebas**: Se determina qué porcentaje de las pruebas (especificado por el parámetro truncation\_percentage) se considera el de menor rendimiento y se cancela.
3. **Continuación del Entrenamiento**: Si la nueva prueba no está en el porcentaje de menor rendimiento, el trabajo de optimización continúa; de lo contrario, se detiene.

**Ejemplo de Implementación**

Aquí tienes un ejemplo de cómo aplicar una política de selección por truncamiento en un trabajo de optimización:

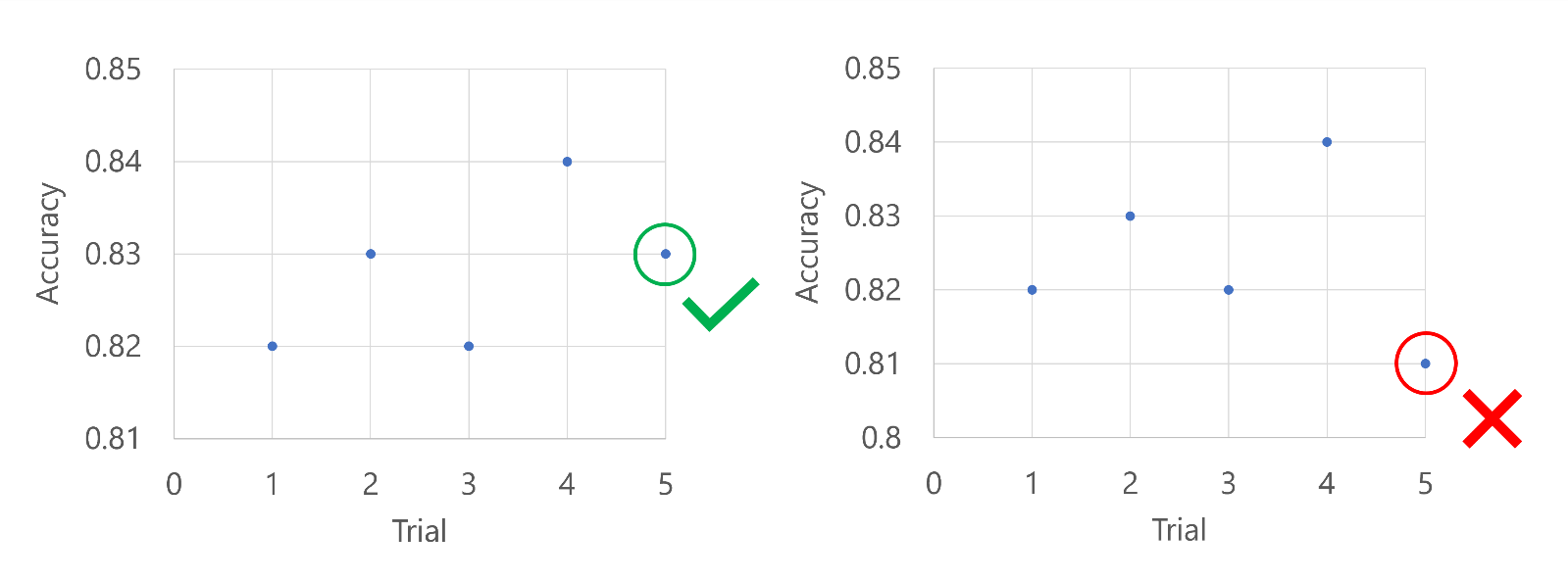
Texto

Descripción generada automáticamente

**Ejemplo de Evaluación**

Imagina que la métrica principal es la precisión del modelo. Cuando se registra la precisión en el quinto trial, la política determinará si esta precisión está entre el 20% de las pruebas de peor rendimiento realizadas hasta ese momento. En este caso, el 20% se traduce en una prueba:

* Si la precisión del quinto trial no está en el grupo de menor rendimiento, la búsqueda de hiperparámetros continuará.
* Si el quinto trial resulta ser el de menor precisión de todos los realizados hasta ahora, el trabajo de optimización se detendrá, evitando así el entrenamiento de modelos que no aportan mejoras.



-- **Uso de un Trabajo de Barrido para la Optimización de Hiperparámetros en Azure Machine Learning**

En Azure Machine Learning, puedes ajustar hiperparámetros ejecutando un trabajo de barrido. Aquí tienes una guía paso a paso sobre cómo configurar y ejecutar un trabajo de barrido para la optimización de hiperparámetros.

**1. Crear un Script de Entrenamiento**

Para ejecutar un trabajo de barrido, necesitas crear un script de entrenamiento que:

* Incluya un argumento para cada hiperparámetro que desees variar.
* Registre la métrica de rendimiento objetivo utilizando MLflow.

**Ejemplo de Script de Entrenamiento**

Texto

Descripción generada automáticamente

**2. Configurar y Ejecutar un Trabajo de Barrido**

A continuación, prepara el trabajo de barrido creando un trabajo de comando que especifique qué script ejecutar y defina los parámetros utilizados por el script:

Texto

Descripción generada automáticamente**Configuración del Trabajo de Comando Base**

Sobrescribir Parámetros de Entrada con el Espacio de Búsqueda

Texto

Descripción generada automáticamente

**3. Llamar a sweep() en tu Trabajo de Comando**

Finalmente, llama al método sweep() en tu trabajo de comando para barrer sobre tu espacio de búsqueda:

Texto

Descripción generada automáticamente

**4. Monitorear y Revisar los Trabajos de Barrido**

Puedes monitorear los trabajos de barrido en el estudio de Azure Machine Learning. El trabajo de barrido iniciará pruebas para cada combinación de hiperparámetros a probar. Para cada prueba, puedes revisar todas las métricas registradas.

Además, puedes evaluar y comparar modelos visualizando las pruebas en el estudio. Puedes ajustar cada gráfico para mostrar y comparar los valores de los hiperparámetros y las métricas para cada prueba, lo que te permitirá identificar el mejor modelo.

--