COLOMBIA POTENCIA DE LA VIDA

TIC

Optimización del modelo

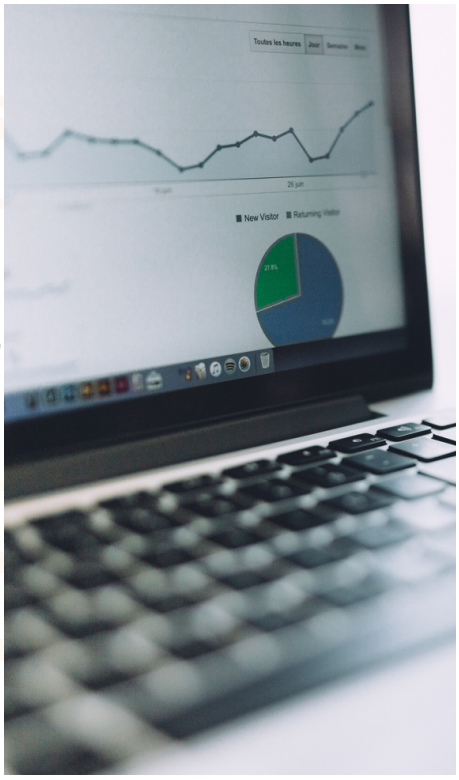
TALENTO TECH

AZ | PROYECTOS EDUCATIVOS

UTP Universidad Tecnológica de Pereira

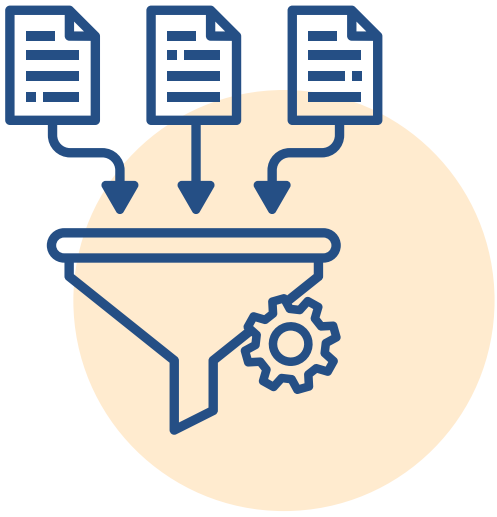
Optimización del modelo

En Optimización del Modelo, nos centraremos en técnicas avanzadas para mejorar el rendimiento y la generalización de nuestros modelos de inteligencia artificial. Exploraremos la afinación de hiperparámetros, donde aprenderemos a ajustar los parámetros del modelo para optimizar su rendimiento. Además, profundizaremos en la validación cruzada, una técnica esencial para evaluar la capacidad de generalización de nuestros modelos y evitar el sobreajuste. Estos subtemas nos proporcionarán las herramientas necesarias para desarrollar modelos más precisos y robustos en una variedad de aplicaciones de aprendizaje automático.



Optimización del Modelo: Afinación de Hiperparámetros y Validación Cruzada

La optimización del modelo es un proceso crucial en el desarrollo de sistemas de inteligencia artificial eficaces. Incluye técnicas como la afinación de hiperparámetros y la validación cruzada para mejorar el rendimiento y la generalización del modelo. A continuación, veremos el detalle de algunas técnicas.



1 Afinación de Hiperparámetros

Los hiperparámetros son configuraciones que no se aprenden directamente del conjunto de datos, pero afectan el comportamiento y el rendimiento del modelo durante el entrenamiento. La afinación de hiperparámetros implica encontrar la combinación óptima de valores para estos parámetros con el fin de mejorar el rendimiento del modelo. Algunas estrategias comunes de afinación de hiperparámetros incluyen:

Búsqueda de Cuadrícula

Consiste en definir una cuadrícula de valores para los hiperparámetros y evaluar el rendimiento del modelo para cada combinación de valores en la cuadrícula. Este enfoque es exhaustivo pero puede ser computacionalmente costoso.

Búsqueda Aleatoria

En lugar de evaluar todas las combinaciones posibles de hiperparámetros, la búsqueda aleatoria selecciona un conjunto aleatorio de combinaciones y evalúa su rendimiento. Esto puede ser más eficiente en términos computacionales y puede encontrar configuraciones óptimas en menos tiempo

2 Validación Cruzada

La validación cruzada es una técnica utilizada para evaluar el rendimiento de un modelo y estimar su capacidad de generalización. Consiste en dividir el conjunto de datos en k subconjuntos llamados "folds", entrenar el modelo en k-1 de estos folds y evaluar su rendimiento en el fold restante. Este proceso se repite k veces, utilizando cada fold como conjunto de prueba exactamente una vez.

Algunas formas comunes de validación cruzada incluyen:

Validación Cruzada K-Fold

Divide el conjunto de datos en k folds y repite el proceso k veces, utilizando cada fold como conjunto de prueba exactamente una vez.

Validación Cruzada Estratificada K-Fold

Similar a la validación cruzada k-fold, pero asegura que cada fold mantenga la misma proporción de clases que el conjunto de datos original.

GridSearchCV

GridSearchCV es una técnica exhaustiva que busca a través de una cuadrícula de hiperparámetros predefinidos y evalúa el rendimiento del modelo para cada combinación posible de valores de hiperparámetros utilizando validación cruzada. Este enfoque es útil cuando el espacio de búsqueda de hiperparámetros no es demasiado grande y se puede explorar de manera sistemática. Aquí hay una descripción paso a paso de cómo usar GridSearchCV:

1 Importar GridSearchCV

Primero, importamos la clase GridSearchCV de la biblioteca sklearn.model_selection.

2 Definir la Cuadrícula de Hiperparámetros

Luego, definimos una cuadrícula de hiperparámetros que queremos explorar. Esto incluye los hiperparámetros del modelo y una lista de valores para cada hiperparámetro que queremos probar.

3 Inicializar el Estimador del Modelo

Después, inicializamos el estimador del modelo que queremos ajustar. Este estimador puede ser cualquier modelo de Scikit-Learn, como un clasificador de bosque aleatorio o una máquina de vectores de soporte.

4 Crear el Objeto GridSearchCV

Creamos un objeto GridSearchCV pasando el estimador del modelo, la cuadrícula de hiperparámetros y la métrica de evaluación que queremos optimizar, como argumentos.

5 Ajustar GridSearchCV

Finalmente, ajustamos el objeto GridSearchCV a nuestros datos utilizando el método fit(), que realiza la búsqueda de la mejor combinación de hiperparámetros mediante validación cruzada.

RandomizedSearchCV

RandomizedSearchCV es una técnica más eficiente que GridSearchCV, especialmente cuando el espacio de búsqueda de hiperparámetros es grande. En lugar de evaluar todas las combinaciones posibles de valores de hiperparámetros, RandomizedSearchCV selecciona un número fijo de configuraciones de hiperparámetros al azar y evalúa su rendimiento utilizando validación cruzada.

Esto permite explorar un espacio de hiperparámetros más grande en menos tiempo. Aquí está cómo usar RandomizedSearchCV:

1 Importar RandomizedSearchCV

Al igual que con GridSearchCV, primero importamos la clase RandomizedSearchCV de la biblioteca sklearn.model_selection.

2 Definir la Distribución de Hiperparámetros

En lugar de una cuadrícula de valores de hiperparámetros, definimos una distribución de probabilidad para cada hiperparámetro que queremos explorar. Esto se hace utilizando las distribuciones proporcionadas por la biblioteca scipy.stats.

3 Inicializar el Objeto RandomizedSearchCV

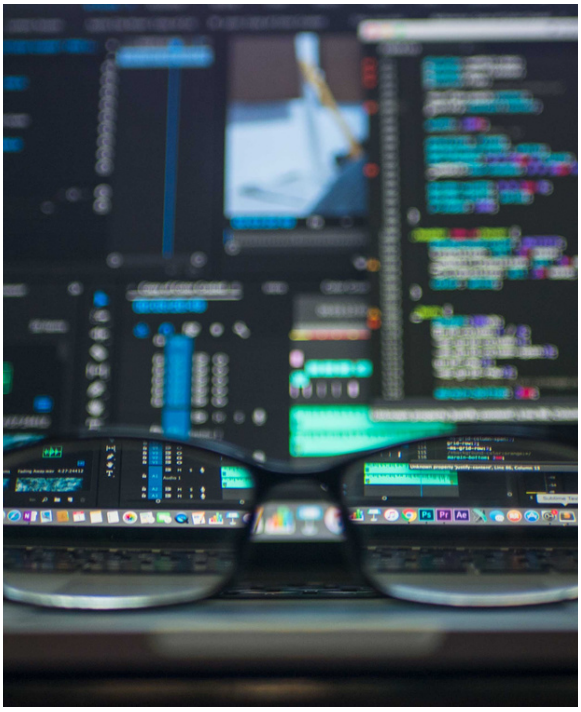
Creamos un objeto RandomizedSearchCV pasando el estimador del modelo, la distribución de hiperparámetros, el número de iteraciones y la métrica de evaluación.

4 Ajustar RandomizedSearchCV

Finalmente, ajustamos el objeto RandomizedSearchCV a nuestros datos utilizando el método fit(), que selecciona aleatoriamente las configuraciones de hiperparámetros y evalúa su rendimiento mediante validación cruzada.

5 ¿Qué es la Validación Cruzada?

La validación cruzada es una técnica usada para evaluar el rendimiento de un modelo de aprendizaje automático utilizando diferentes subconjuntos de datos para entrenamiento y prueba. En lugar de dividir el conjunto de datos en un único conjunto de entrenamiento y prueba, la validación cruzada divide el conjunto de datos en múltiples subconjuntos o "folds". Luego, entrena y evalúa el modelo varias veces, utilizando diferentes combinaciones de subconjuntos de entrenamiento y prueba en cada iteración.



6 Procedimiento de Validación Cruzada

Finalmente, ajustamos el objeto RandomizedSearchCV a nuestros datos utilizando el método fit(), que selecciona aleatoriamente las configuraciones de hiperparámetros y evalúa su rendimiento mediante validación cruzada.

a. División en k Folds

El conjunto de datos se divide en k subconjuntos o "folds" de aproximadamente igual tamaño.

b. Iteraciones

Se repite el proceso k veces, utilizando cada fold como conjunto de prueba exactamente una vez y los restantes como conjunto de entrenamiento.

c. Evaluación del Modelo

En cada iteración, se entrena el modelo en el conjunto de entrenamiento y se evalúa su rendimiento en el conjunto de prueba. Se registra el rendimiento del modelo en cada iteración.

d. Promedio de resultados

Finalmente, se calcula el promedio de los resultados de todas las iteraciones para obtener una estimación más precisa del rendimiento del modelo.

7 Importancia en la Evaluación del Modelo

La validación cruzada es crucial en la evaluación del modelo por:

a. Estimación más precisa del rendimiento: Al promediar los resultados de múltiples iteraciones, la validación cruzada proporciona una estimación más precisa del rendimiento del modelo en datos no vistos.

b. Reducción del Sesgo de Evaluación: Al usar diferentes combinaciones de subconjuntos de entrenamiento y prueba, la validación cruzada reduce el sesgo en la evaluación del modelo y proporciona una evaluación más objetiva.

c. Detección de Sobreajuste: La validación cruzada nos permite detectar si nuestro modelo está sobreajustando los datos de entrenamiento al evaluar su rendimiento en múltiples subconjuntos de datos de prueba.

8 Tipos de Validación Cruzada

a. Validación Cruzada K-Fold

Divide el conjunto de datos en k folds y repite el proceso k veces, utilizando cada fold como conjunto de prueba exactamente una vez.

b. Validación Cruzada Estratificada K-Fold

Similar a la validación cruzada k-fold, pero asegura que cada fold mantenga la misma proporción de clases que el conjunto de datos original.

c. Validación Cruzada Leave-One-Out (LOO)

Cada muestra se utiliza como conjunto de prueba una vez, mientras que el resto se utiliza como conjunto de entrenamiento.

d. Validación Cruzada Aleatoria

Selecciona aleatoriamente subconjuntos de entrenamiento y prueba en cada iteración.