Sesión 4: Algoritmos de Clasificación y Métricas de Desempeño en Clasificación

En esta sesión exploraremos los algoritmos de clasificación, centrándonos en K-Nearest Neighbors (K-NN), y analizaremos las diferentes métricas para evaluar el desempeño de modelos de clasificación.

R **por Kibernum Capacitación S.A. Kibernum Capacitación S.A.**

Algoritmo K-Nearest Neighbors (K-NN)

K-Nearest Neighbors (K-NN) es uno de los algoritmos más intuitivos y fáciles de implementar en problemas de clasificación. Pertenece a la familia de algoritmos de aprendizaje supervisado y se basa en la idea de que los datos similares suelen estar cerca unos de otros en el espacio de características.

En lugar de construir un modelo durante la fase de entrenamiento, K-NN simplemente memoriza el conjunto de datos. Luego, para clasificar una nueva instancia, busca los "vecinos" más cercanos entre los ejemplos conocidos y asigna la clase más común entre ellos.

K-NN es un algoritmo perezoso (lazy learner): no realiza ningún tipo de abstracción del conjunto de datos hasta el momento de la predicción.

Clasificar por cercanía: así piensa K-NN

Esta imagen muestra de manera clara el flujo de trabajo del algoritmo K-Nearest Neighbors (K-NN):

- Primero se identifican los datos existentes etiquetados (estrellas rojas = Clase A, triángulos verdes = Clase B).
- Luego se calcula la distancia entre el nuevo dato (signo de interrogación) y los ejemplos conocidos.
- Finalmente, se seleccionan los k vecinos más cercanos (en este caso, k=3) y se vota por mayoría para asignar la clase más probable.

K-NN no "aprende" en el entrenamiento, sino que decide en el momento de la predicción, comparando con su entorno.

Este enfoque lo hace muy útil cuando la distribución de datos es compleja o no lineal, y donde la proximidad es un buen predictor.

Pasos principales del algoritmo K-NN

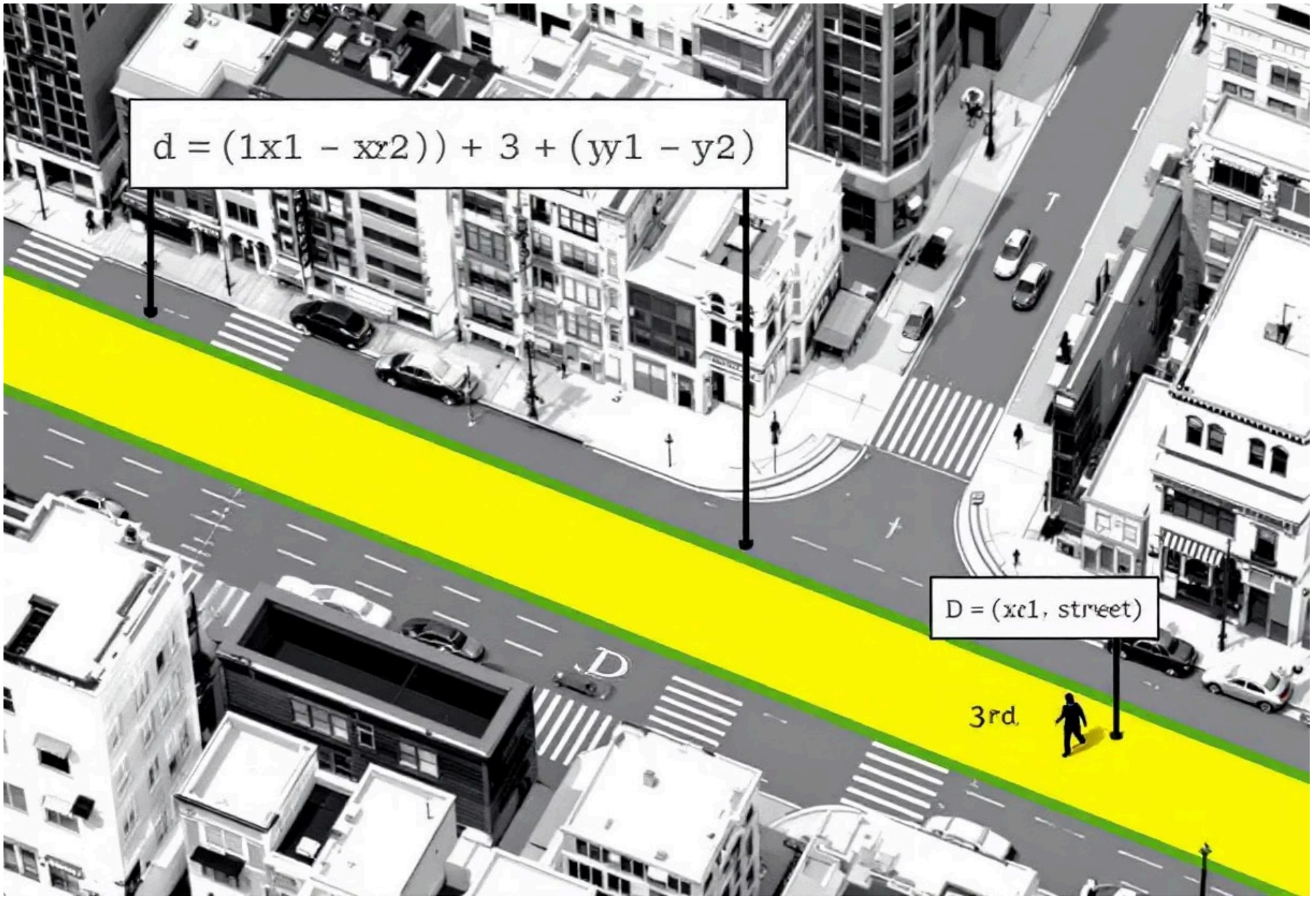
1. Elegir el valor de K (número de vecinos más cercanos a considerar).
2. Calcular la distancia entre el nuevo punto y todos los ejemplos del conjunto de entrenamiento.
3. Seleccionar los K vecinos más cercanos.
4. Contar la frecuencia de las clases entre estos vecinos.
5. Asignar la clase mayoritaria al nuevo ejemplo.

Distancias comunes en K-NN

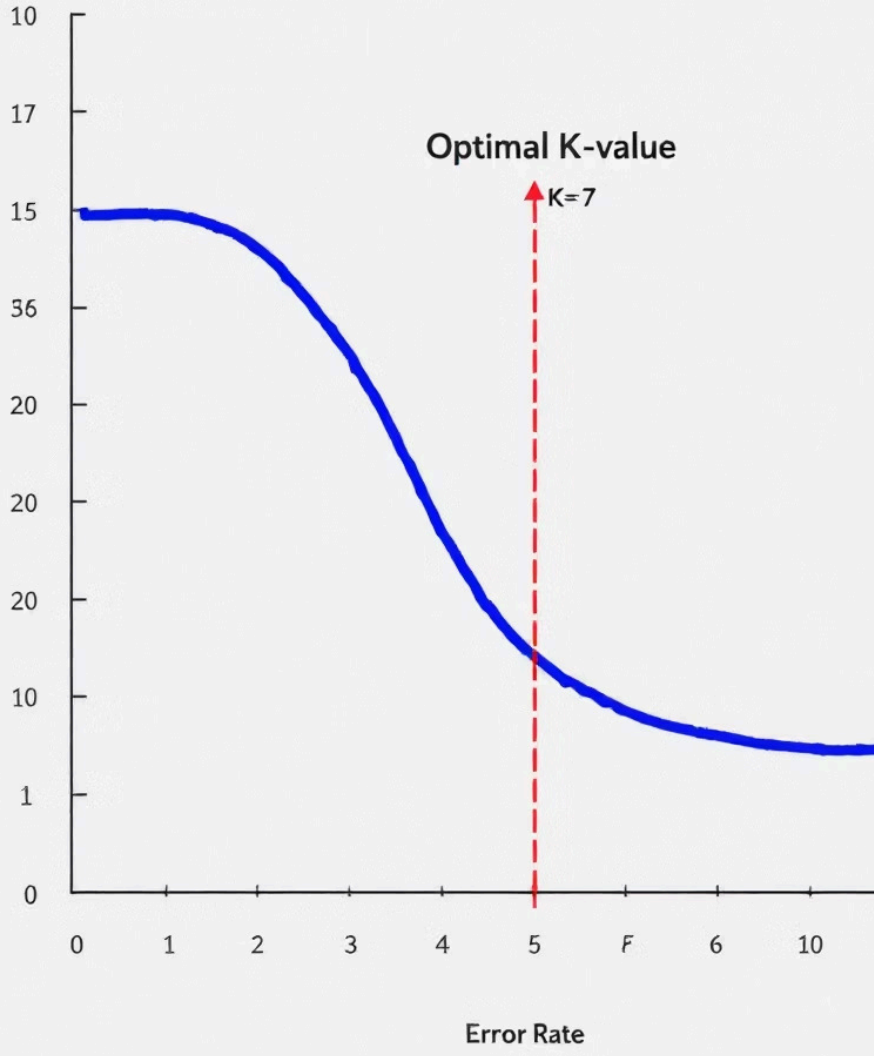
El desempeño del algoritmo depende del cálculo de la distancia entre puntos. Algunas de las distancias más comunes son:

Distancia Euclidiana (por defecto en Scikit-learn):

Distancia Manhattan:



Estas métricas se ven afectadas por la escala de las variables, por lo que es importante normalizar o estandarizar los datos antes de aplicar K-NN.



Elección del valor de K

El valor de K (número de vecinos a considerar) es un parámetro crucial que afecta significativamente la precisión del modelo.

- Si K es muy bajo (por ejemplo, 1), el modelo es muy sensible al ruido en los datos → sobreajuste (overfitting).
- Si K es muy alto, se consideran demasiados vecinos, diluyendo las diferencias entre clases → subajuste (underfitting).

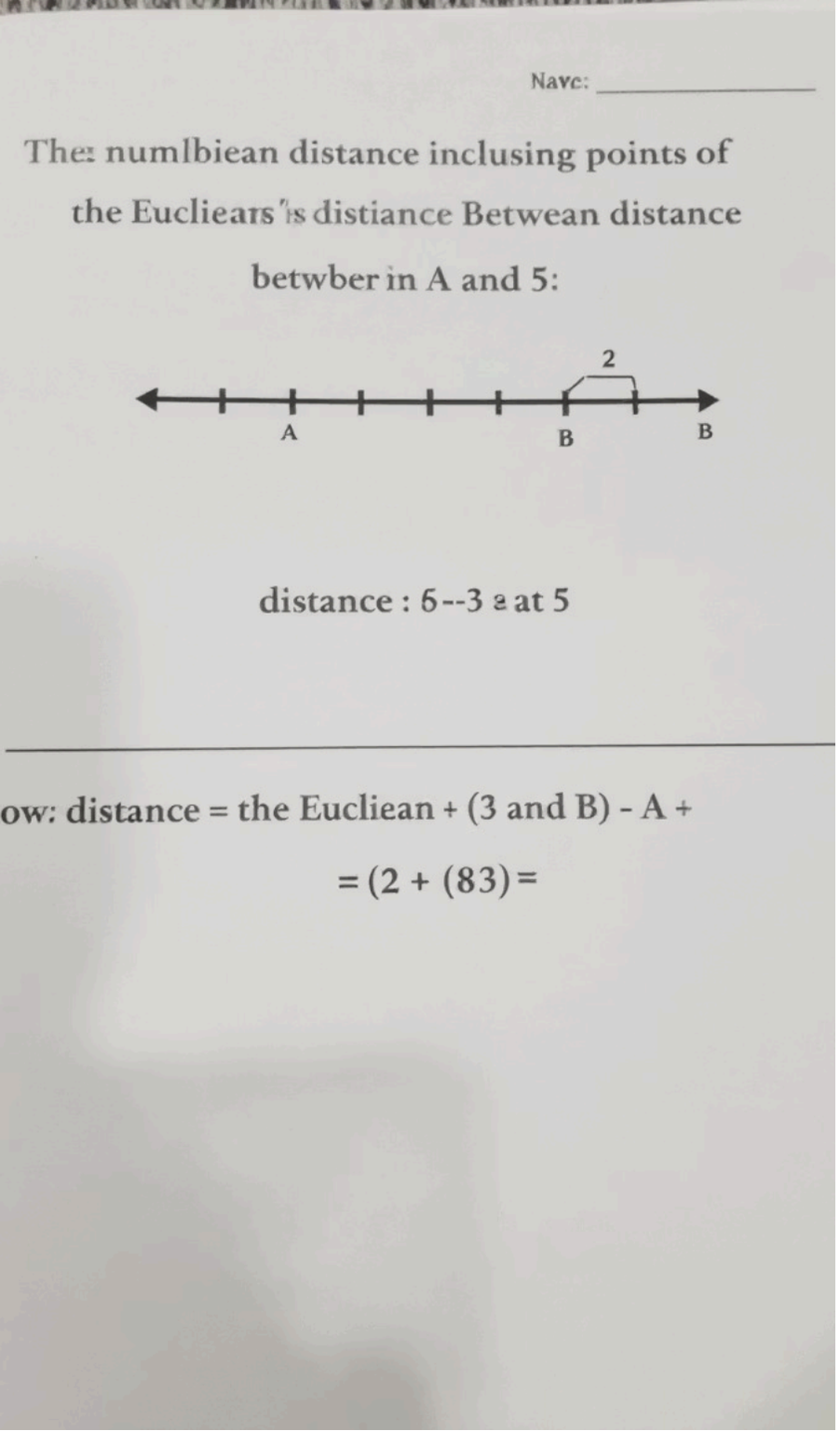
Una práctica común es probar distintos valores de K y graficar el error de validación para elegir el óptimo.

Impacto del valor de K: Un ejemplo realista

Para comprender cómo el valor de K afecta al comportamiento del algoritmo K-Nearest Neighbors, exploraremos un caso concreto y fácil de visualizar: "predecir si un estudiante aprobará un examen en función de las horas que estudió."

Horas de estudio	¿Aprobó el examen?
1 hora	No
2 horas	No
3 horas	Sí
4 horas	Sí
5 horas	Sí
6 horas	Sí
7 horas	No

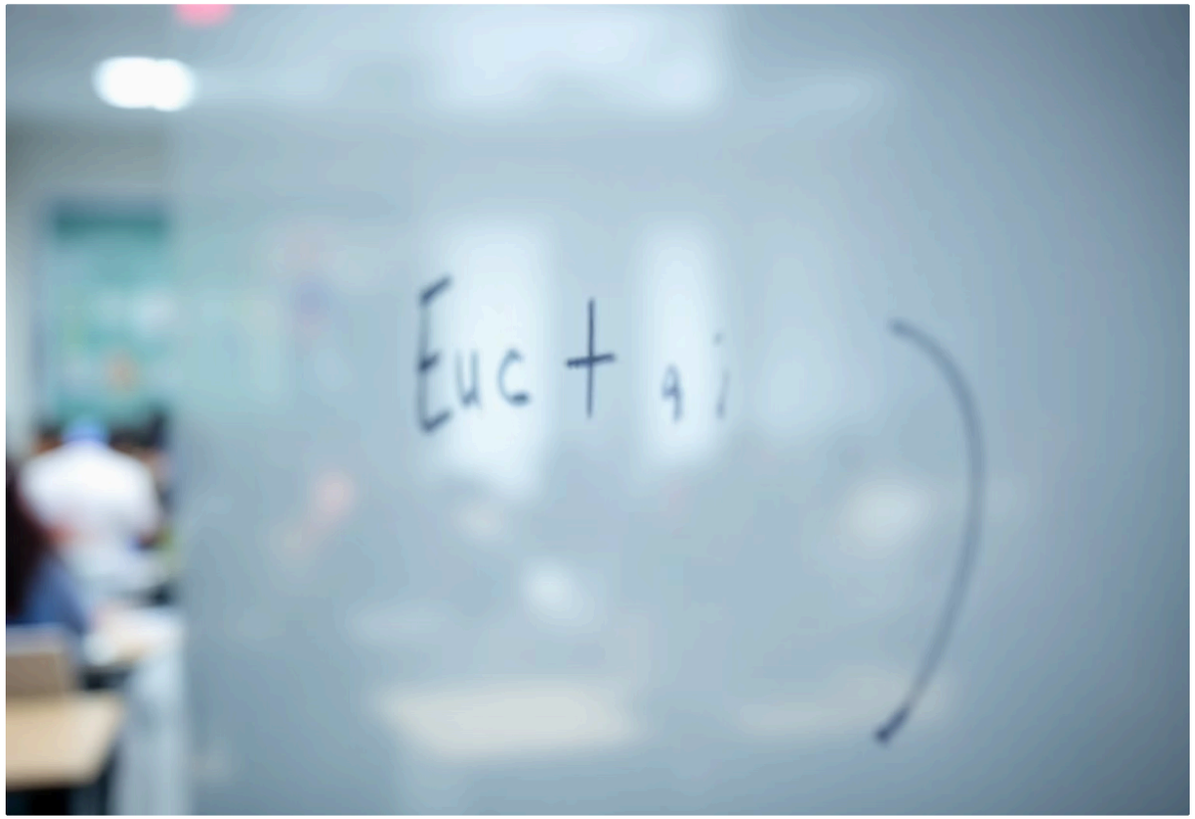
Nuestro objetivo es predecir si un nuevo estudiante, que estudió 4.5 horas, logrará aprobar.



¿Cómo funciona K-NN en este contexto?

K-NN clasifica al nuevo estudiante en función de los vecinos más cercanos (es decir, los más próximos en cantidad de horas estudiadas). Para determinar esa cercanía, usamos la distancia euclidiana simple:

Donde:



Una vez calculadas todas las distancias, se ordenan de menor a mayor y se seleccionan los K vecinos más cercanos.

Comparando dos valores de K: Caso 1 (K = 1)

Sobreajuste (Overfitting)

- Se considera solo el vecino más cercano.
- En este caso, el estudiante que estudió 5 horas y aprobó es el más próximo.
- El modelo predice: "Sí, aprobará".

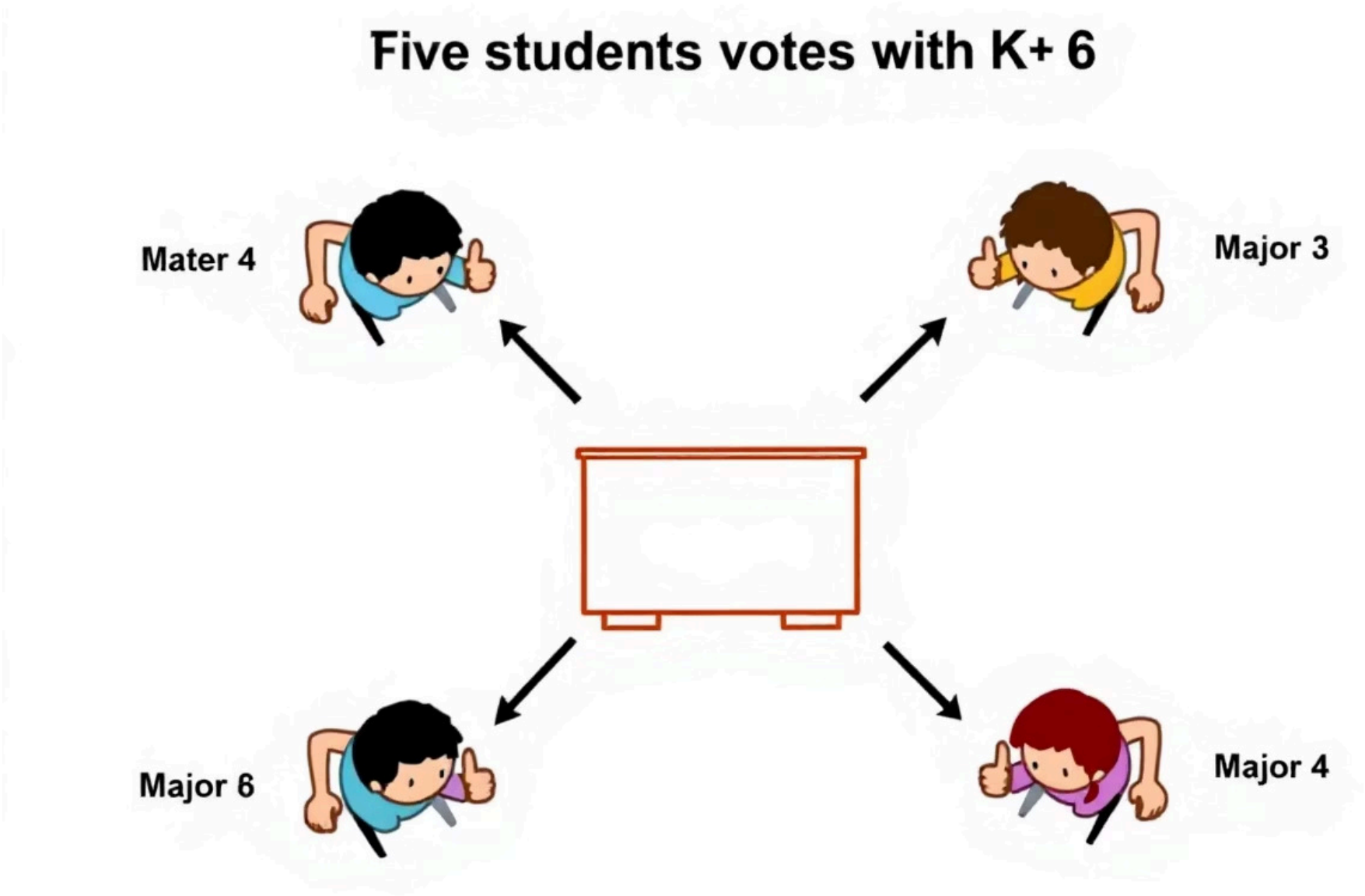
Riesgo: Si ese único vecino estuviera mal etiquetado, afectaría gravemente la predicción. Esto hace que el modelo sea demasiado sensible al ruido, es decir, susceptible a errores de clasificación por depender de un solo ejemplo.

Este es un caso clásico de sobreajuste, donde el modelo se adapta tanto al detalle que pierde capacidad de generalizar.

Comparando dos valores de K: Caso 2 (K = 6)

Subajuste (Underfitting)

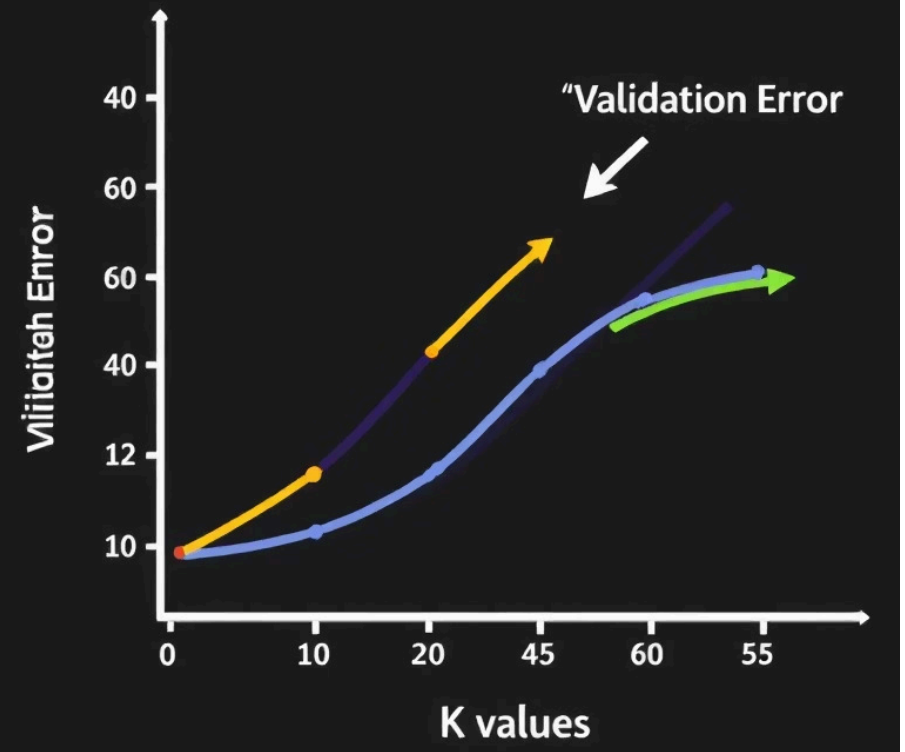
- Se consideran los 6 estudiantes más cercanos.
- De ellos, 4 aprobaron y 2 no aprobaron.



El modelo aún predice: "Sí, aprobará", por mayoría.

Pero si el valor de K aumentara a 7, todos los puntos serían considerados, incluyendo a estudiantes con solo 1 o 2 horas de estudio que no aprobaron.

Esto hace que el modelo pierda sensibilidad a los límites entre clases y generalice en exceso. Esto se conoce como subajuste.



¿Cómo encontrar el valor óptimo de K?

Una práctica común es graficar el error de validación para diferentes valores de K, y observar en qué punto se minimiza.

Esto se conoce como el método del "codo" (elbow). La idea es elegir un valor de K lo suficientemente bajo como para detectar patrones, pero no tan bajo que se sobreajuste.

En este caso, K = 3 o K = 5 podrían ser candidatos razonables, al ofrecer un equilibrio entre precisión y robustez.

Evaluación de precisión en K-NN

Ya sabemos cómo funciona el algoritmo K-Nearest Neighbors (K-NN) y cómo elegir el valor de K. Ahora, es momento de evaluar si el modelo realmente está clasificando bien.

La evaluación se realiza utilizando métricas de clasificación, las cuales nos permiten entender:

- ¿Cuántas predicciones fueron correctas?
- ¿Qué clase se confunde más?
- ¿El modelo tiene algún sesgo?

Principales métricas en clasificación

Precisión (Accuracy)

Porcentaje de predicciones correctas sobre el total de observaciones. Es útil como referencia general, pero puede ser engañosa si una clase domina el conjunto de datos.

Matriz de Confusión

Muestra cuántos elementos de cada clase fueron correctamente clasificados (diagonal) y cuántos fueron confundidos con otra clase. Es una herramienta fundamental para analizar errores específicos por clase.

Sensibilidad (Recall)

También llamada tasa de verdaderos positivos. Mide la capacidad del modelo para encontrar todos los casos reales de una clase. Muy útil en problemas donde lo importante es detectar una condición (ej. diagnósticos médicos).

Especificidad

Complemento de la sensibilidad. Mide la capacidad del modelo para evitar falsos positivos (es decir, no confundir otras clases con la clase objetivo).

Curva ROC-AUC

Se aplica en clasificación binaria, y permite analizar el balance entre sensibilidad y especificidad en diferentes umbrales. El área bajo la curva (AUC) es una medida global de rendimiento del modelo.

Ejemplo guiado: Clasificación con K-NN usando el dataset Iris

Vamos a implementar y evaluar un modelo K-NN con k=5, utilizando el dataset Iris, que contiene 150 registros de flores de tres especies: setosa, versicolor y virginica.

1. Cargar y preparar los datos

Primero, cargamos el dataset, lo dividimos en conjunto de entrenamiento y prueba, y estandarizamos las características.

La estandarización es importante para que todas las variables tengan la misma escala, lo cual mejora el rendimiento del modelo K-NN.

Entrenamiento y evaluación del modelo K-NN

2. Entrenar el modelo K-NN

Entrenamos el clasificador usando k = 5 vecinos.

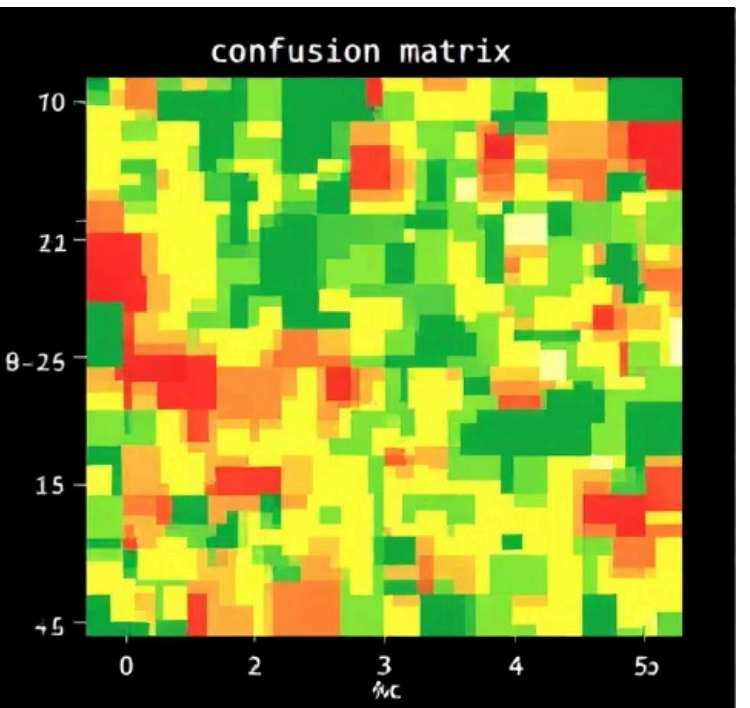
El modelo ahora está listo para predecir la clase de nuevas flores basándose en sus características.

3. Evaluar el modelo

Realizamos las predicciones y aplicamos las métricas de evaluación.

```
turilu iry confusion mata tyine "hesast ario nonarie(()
callssization Iris a irs;
frime iris detrians of an vat(mtain masuil repatx);
    Ir& trallional mesants
    irs classs,
    {
      Irisclass;
    Fur iris detalt ]
    Sterrioh "lin,in cour")
    Conlustisiin "udtal /lecam""
    Rercuris
    Dustical mattion, wutwc(une (F1-scor))
    Conlucsiionf delles, (s)
    ris inar
    Coclurtiant ires
    Nutmporte dintito)
    Exasch betaur of issul in the sectuto "F1-scores"
    pryite w "1ll;

                  recar          msaal   Fis
    precision      A             2       1860
    support wite, F1score        1       683
    Suppport       support;      1       1
    6ul support    8             1       25%/
    Donvination:   f1class       1       25.0
```



Visualización de la Matriz de Confusión

Podemos representar gráficamente la matriz de confusión para entender mejor los aciertos y errores por clase.

Observando los resultados podemos concluir que:

- Precisión global: alrededor del 95%.
- La matriz de confusión debería mostrar valores altos en la diagonal, lo que indica aciertos.
- El reporte de clasificación debería mostrar valores altos de precisión y recall por clase.

Estos resultados confirman que el modelo generaliza bien y clasifica correctamente la mayoría de los casos, con mínimos errores entre las clases versicolor y virginica, que suelen ser más difíciles de distinguir.

Métricas de desempeño en clasificación

Una vez entrenado un modelo de clasificación, necesitamos evaluar su rendimiento de forma cuantitativa. Las siguientes métricas permiten diagnosticar con precisión cómo se comporta el modelo, especialmente en contextos donde ciertas clases son más importantes que otras.

Matriz de Confusión

La matriz de confusión es una herramienta fundamental para analizar cómo se distribuyen los aciertos y errores del modelo. Se utiliza especialmente en problemas de clasificación binaria o multiclase.

Cada fila representa la clase real, y cada columna la clase predicha.

	Predicho: Positivo	Predicho: Negativo
Real: Positivo	Verdadero Positivo (TP)	Falso Negativo (FN)
Real: Negativo	Falso Positivo (FP)	Verdadero Negativo (TN)

Esta tabla permite calcular métricas como precisión, sensibilidad, especificidad y muchas otras.

Precisión (Accuracy) y Sensibilidad (Recall)

Precisión (Accuracy)

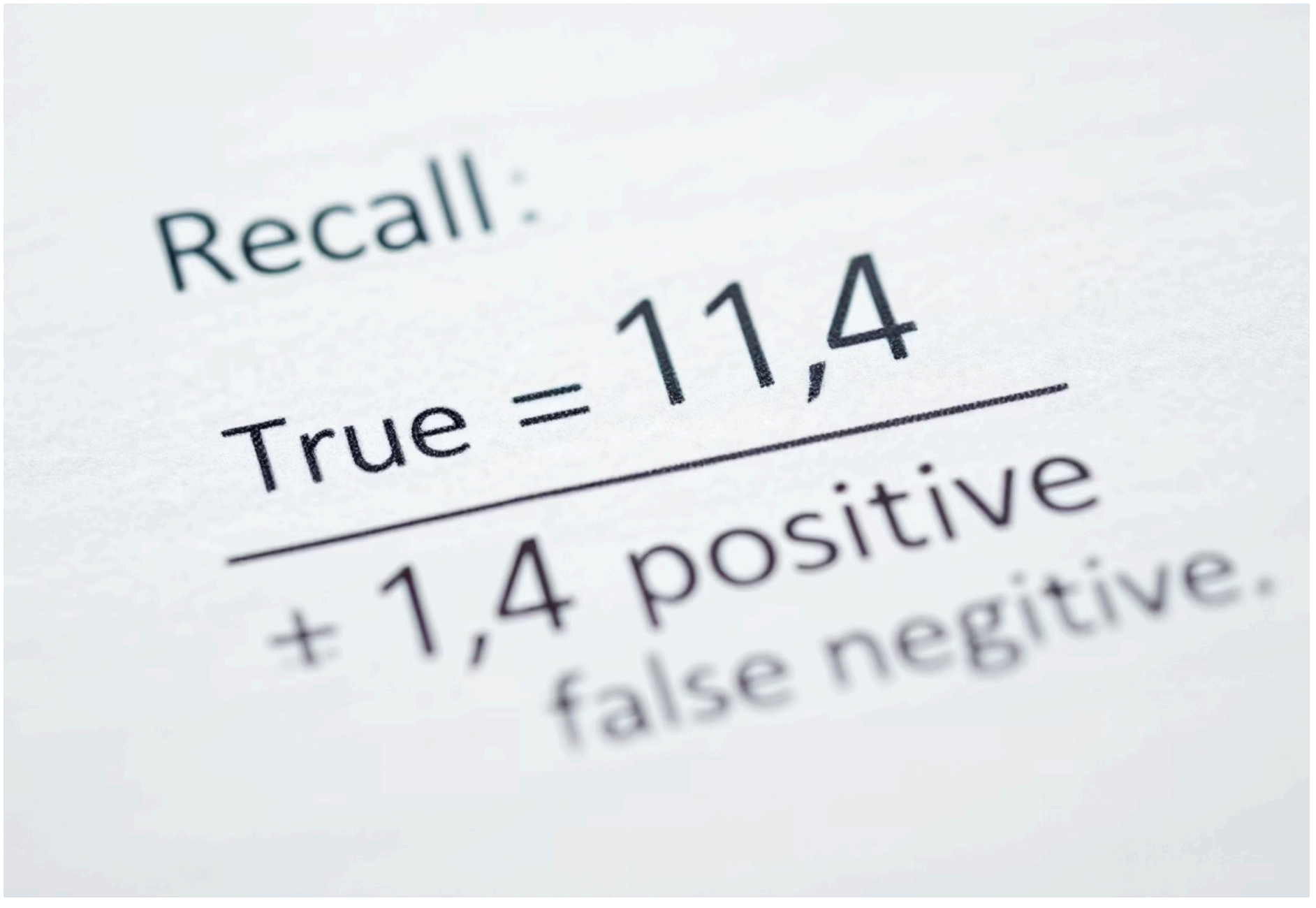
La precisión es una medida global que indica el porcentaje de predicciones correctas realizadas por el modelo.

Esta métrica es útil cuando las clases están equilibradas. Sin embargo, puede ser engañosa si una clase domina el conjunto de datos (por ejemplo, detectar fraudes donde los casos positivos son muy escasos).

Por ejemplo, si el 95% de los datos pertenecen a la clase "No fraude", un modelo que siempre predice "No fraude" tendrá 95% de precisión… pero nunca detectará un caso de fraude real.

Sensibilidad (Recall o Tasa de Verdaderos Positivos)

La sensibilidad mide qué tan bien el modelo identifica los casos positivos reales. También se conoce como recall o tasa de verdaderos positivos.



Se utiliza en situaciones donde es más costoso pasar por alto un caso positivo (por ejemplo, diagnósticos médicos, detección de fraudes o alertas tempranas en sistemas de seguridad).

Especificidad y Curva ROC-AUC

Especificidad (Tasa de Verdaderos Negativos)

La especificidad mide la capacidad del modelo para identificar correctamente los casos negativos. También se conoce como tasa de verdaderos negativos.

Es útil cuando queremos minimizar los falsos positivos, por ejemplo, en sistemas de detección de spam, evaluaciones crediticias, o herramientas de justicia predictiva.

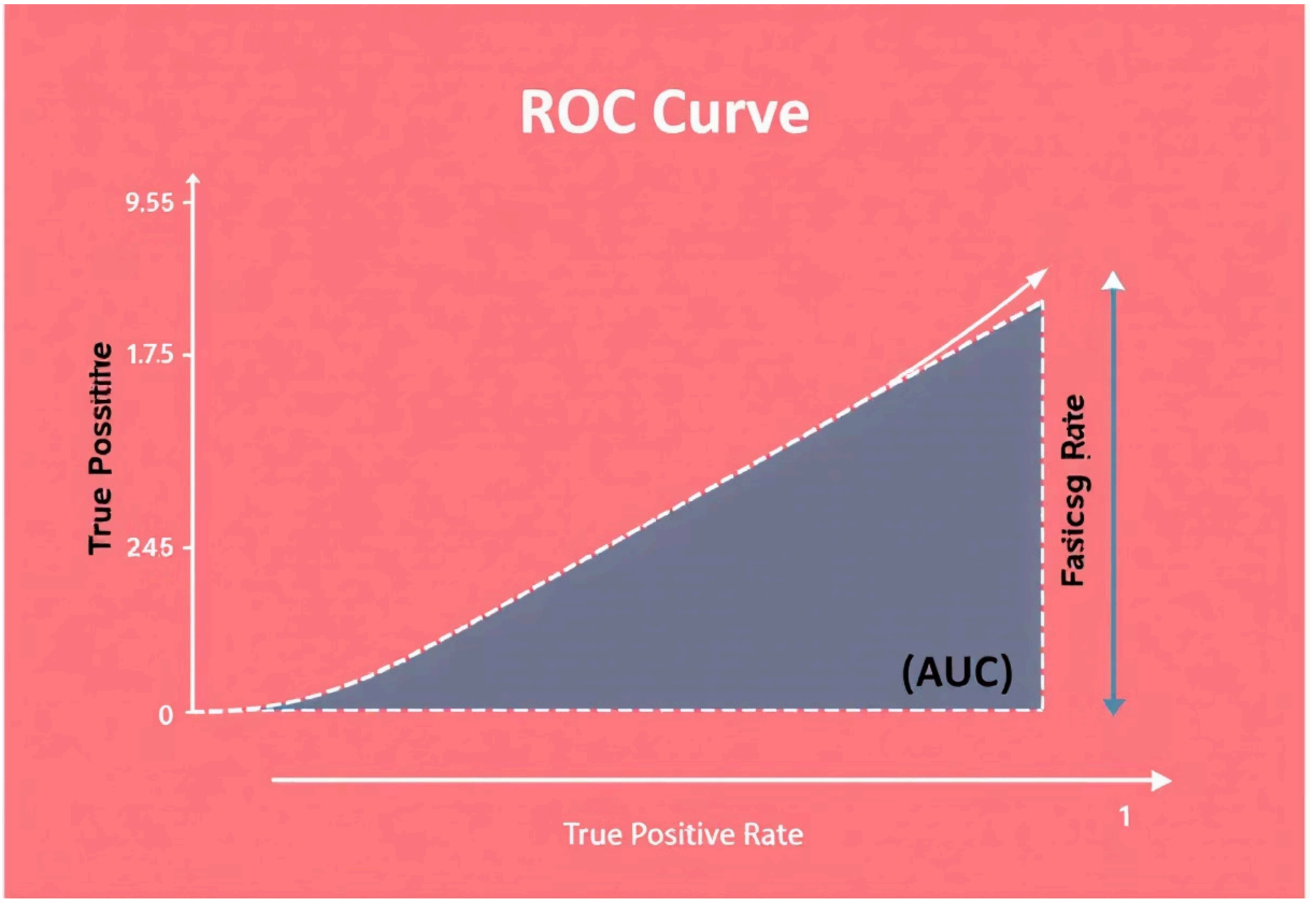
Una alta especificidad significa que el modelo no suele confundir un caso negativo con uno positivo. Es decir, no "alarma" cuando no debe hacerlo.

Curva ROC y AUC

La Curva ROC (Receiver Operating Characteristic) es una herramienta visual que permite analizar el desempeño de un modelo de clasificación binaria bajo distintos umbrales de decisión.

Cada punto de la curva representa una combinación de:

- TPR (Tasa de verdaderos positivos o sensibilidad)
- FPR (Tasa de falsos positivos)



Ejemplo guiado: Evaluación de métricas de clasificación

En este ejemplo trabajaremos con un conjunto de datos simulado que representa pacientes evaluados por un sistema de diagnóstico médico. El objetivo es comprender cómo se calculan e interpretan métricas de evaluación como precisión, sensibilidad, especificidad y valor predictivo positivo, a partir de una matriz de confusión.

Paso 1: Simular el dataset

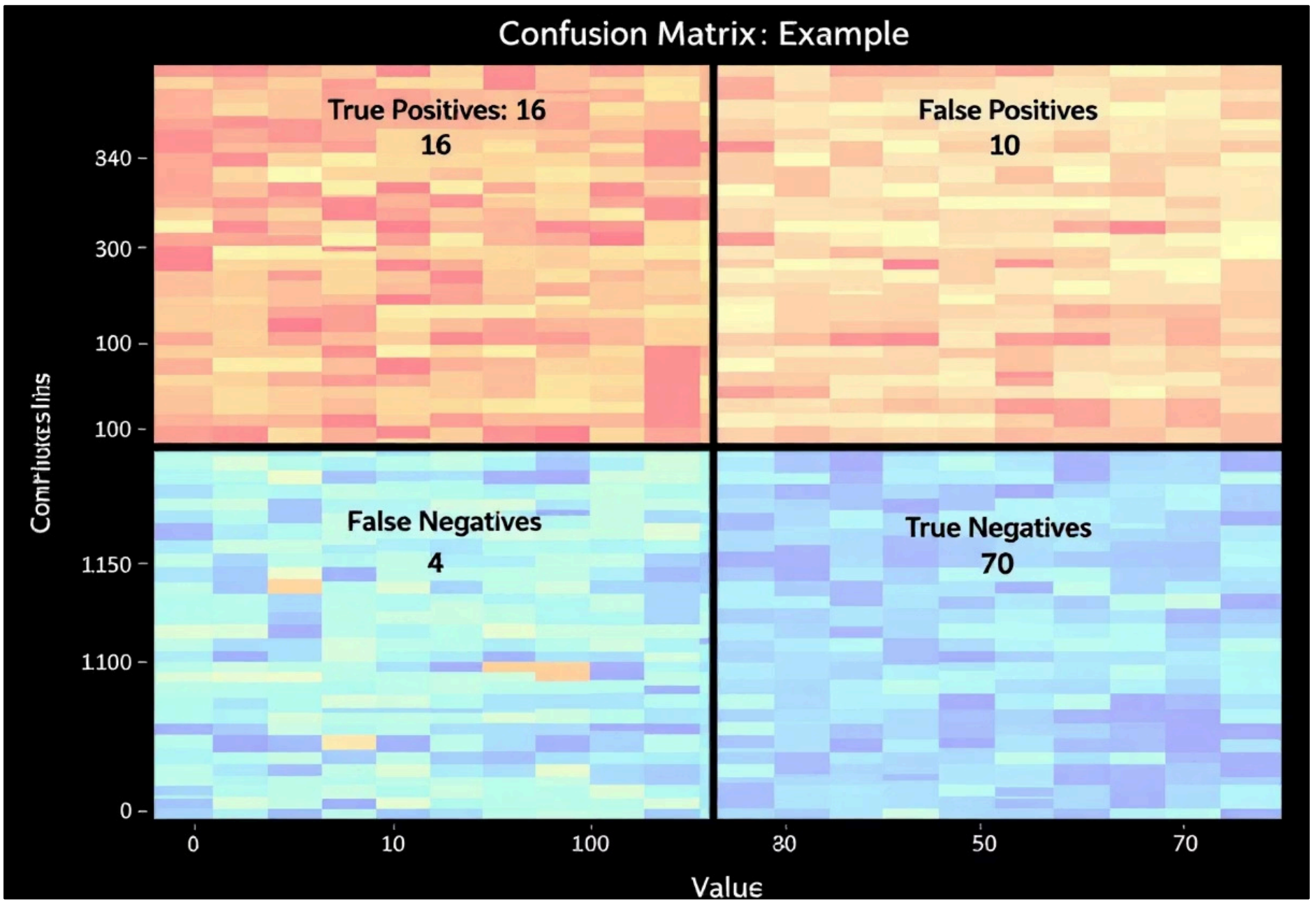
Simulamos 100 pacientes con su diagnóstico real y la predicción hecha por el modelo.

¿Qué representa este dataset?

- 16 pacientes enfermos fueron correctamente diagnosticados (VP).
- 4 pacientes enfermos fueron diagnosticados como sanos (FN).
- 70 pacientes sanos fueron correctamente identificados (VN).
- 10 pacientes sanos fueron diagnosticados erróneamente como enfermos (FP).

Paso 2: Construir la matriz de confusión

Contamos los valores manualmente para calcular la matriz de confusión:



Paso 3: Calcular las métricas

Aplicamos las fórmulas correspondientes para evaluar el modelo:

- Precisión (accuracy): 86.00% → el modelo acierta en 86 de cada 100 pacientes.
- Sensibilidad (recall): 80.00% → identifica correctamente al 80% de los enfermos.
- Especificidad: 87.50% → evita confundir al 87.5% de los pacientes sanos.
- Valor predictivo positivo (precision): 61.54% → de los predichos como enfermos, solo el 61.5% realmente lo está.