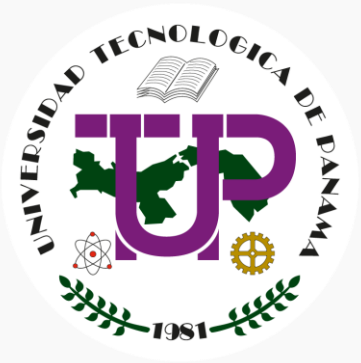


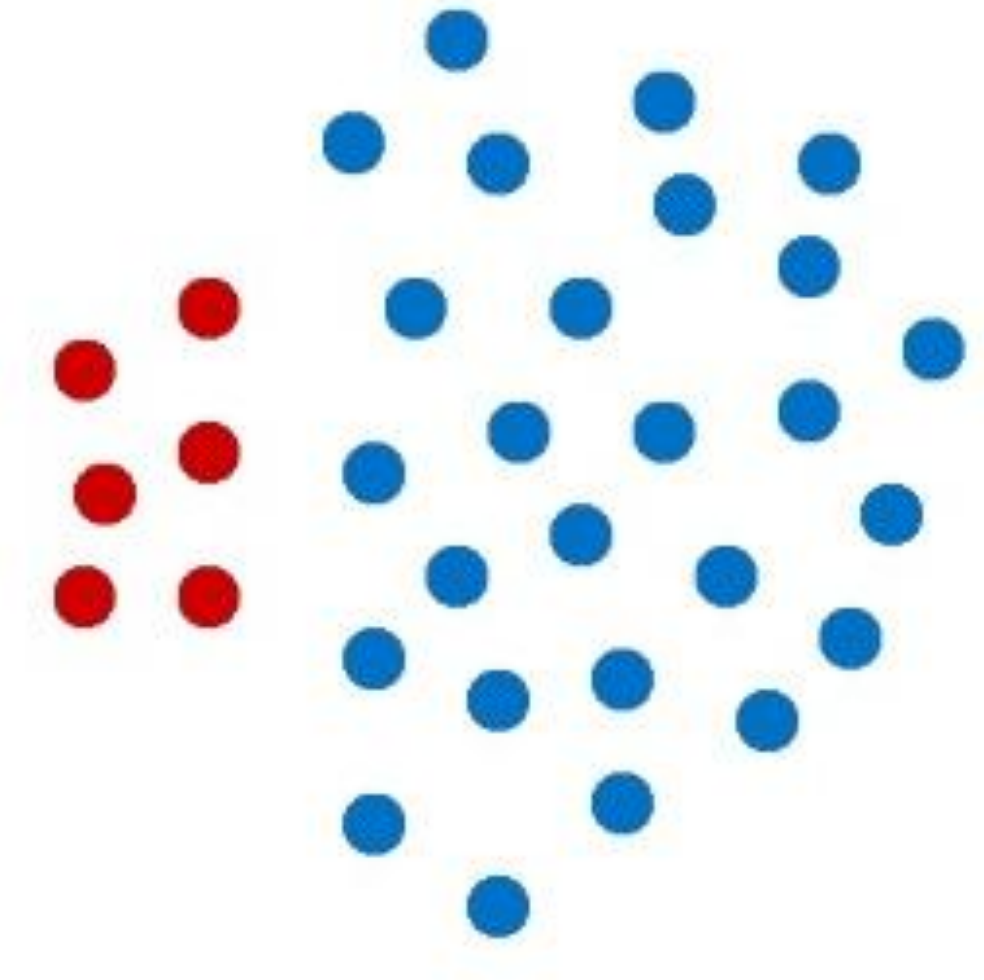
Universidad Tecnológica de Panamá
Facultad de Ingeniería Eléctrica
Maestría en Ingeniería Eléctrica

Semana 2

Introducción a Machine Learning

Balanceo de Clases

El balanceo de clases es muy importante en problemas de clasificación para garantizar que el modelo no esté sesgado hacia la clase mayoritaria.



Importancia del Balanceo de clases

En problemas de clasificación, un desbalance significativo entre clases puede llevar a un modelo que favorezca la clase mayoritaria.

Esto puede resultar en un desempeño deficiente en la predicción de la clase minoritaria.

Consecuencias del desbalance de clases

- Reducción en la capacidad del modelo para generalizar.
- Métricas como la precisión pueden ser engañosas, ya que un modelo puede predecir siempre la clase mayoritaria y obtener una alta precisión.

Estrategias communes para resolver el imbalance de clases

Sobremuestreo de clases minoritarias

- Consiste en aumentar artificialmente el número de ejemplos de la clase minoritaria. Esto puede hacerse duplicando ejemplos existentes o generando nuevos ejemplos sintéticos.
- El objetivo es equilibrar la proporción de clases para que el modelo no esté sesgado hacia la clase mayoritaria.
- A este concepto se le conoce como *data augmentation*, el cual es ampliamente utilizado en problemas de machine learning para mejorar la diversidad de los datos y aumentar la capacidad del modelo para generalizar.

Sobremuestreo de clases minoritarias

- **Ventajas:** Permite aprovechar toda la información disponible de la clase mayoritaria y mejora la capacidad del modelo para aprender patrones de la clase minoritaria.
- **Desventajas:** Puede llevar a sobreajuste si se duplican demasiados ejemplos o si los ejemplos sintéticos no representan bien la distribución real de los datos.
- **Ejemplo práctico:** Si en un conjunto de datos de fraude bancario solo el 5% de las transacciones son fraudulentas, se pueden generar ejemplos adicionales de fraude para que ambas clases tengan una representación similar.

Estrategias communes para resolver el imbalance de clases

Submuestreo de clases mayoritarias

- Esta técnica reduce el número de ejemplos de la clase mayoritaria, seleccionando aleatoriamente un subconjunto de sus datos. Así, se iguala el número de ejemplos entre clases.
- Es una estrategia útil para problemas donde la clase mayoritaria domina significativamente el conjunto de datos, lo que puede dificultar que el modelo aprenda patrones relevantes de la clase minoritaria.
- Al aplicar esta técnica, es fundamental garantizar que los datos restantes de la clase mayoritaria sean representativos y mantengan la diversidad necesaria para el entrenamiento del modelo.

Estrategias communes para resolver el imbalance de clases

- **Ventajas:** Reduce el tamaño del conjunto de datos, lo que puede disminuir el tiempo de entrenamiento.
- **Desventajas:** Existe el riesgo de perder información valiosa al eliminar ejemplos de la clase mayoritaria, lo que puede afectar la capacidad predictiva del modelo.
- **Ejemplo práctico:** En el mismo caso de fraude bancario, se podrían eliminar transacciones legítimas hasta igualar el número de transacciones fraudulentas.

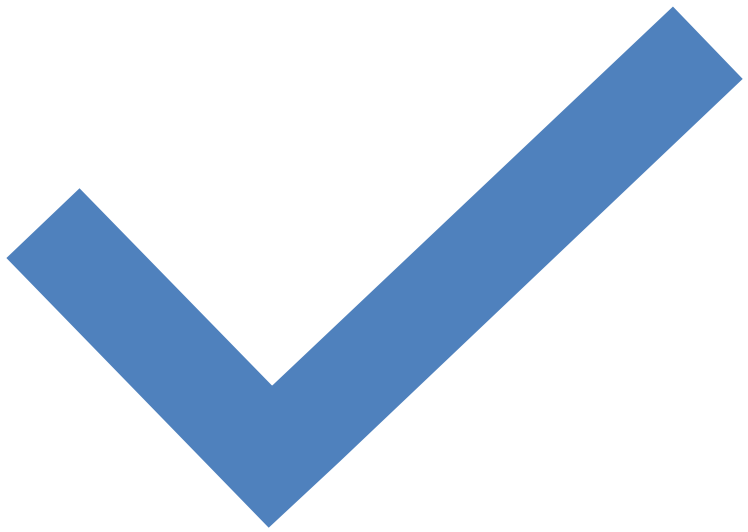
Estrategias communes para resolver el imbalance de clases

Generación sintética de ejemplos (SMOTE y variantes)

Técnicas como SMOTE (Synthetic Minority Over-sampling Technique) crean nuevos ejemplos sintéticos de la clase minoritaria interpolando entre ejemplos reales. Esto ayuda a crear una frontera de decisión más generalizada para el modelo.

Generación sintética de ejemplos (SMOTE y variantes)

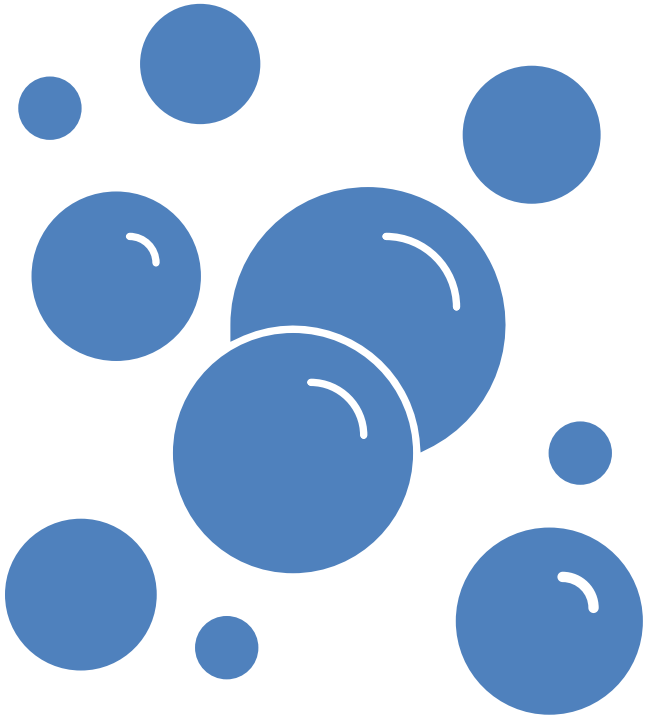
- **Ventajas:** Genera ejemplos más variados que el simple duplicado, ayudando a reducir el sobreajuste.
- **Desventajas:** Puede generar ejemplos poco realistas si los datos originales son muy dispersos o si hay ruido.
- **Ejemplo práctico:** SMOTE se utiliza frecuentemente en problemas de clasificación médica donde las clases minoritarias (por ejemplo, presencia de una enfermedad rara) son de especial interés.



Consideraciones al aplicar estas técnicas

- Es importante evaluar el impacto de estas estrategias en el desempeño del modelo utilizando métricas adecuadas (no solo accuracy).
- Se recomienda realizar validación cruzada para evitar sobreajuste.
- La elección de la técnica depende del contexto y de la cantidad de datos disponibles.

Métricas de evaluación



Métricas de Clasificación

Matriz de confusión

En los problemas de clasificación supervisada, el objetivo es predecir una clase a partir de un conjunto de características. La predicción se compara con el valor real esperado para determinar si el modelo acertó o se equivocó.

La matriz de confusión es una herramienta fundamental para evaluar el desempeño de los algoritmos de clasificación, ya que permite analizar los aciertos y errores del modelo de manera detallada.

La matriz de confusión se construye a partir de los resultados de cada clasificación a nivel individual.

Por ejemplo, supongamos que tenemos la siguiente tabla de resultados de un clasificador binario.

En esta tabla, se muestran 10 instancias con el número de muestra, el resultado esperado (real) y el resultado estimado por el algoritmo.

Número de muestra	Resultado esperado	Resultado estimado
1	1	1
2	0	0
3	1	0
4	0	1
5	1	1
6	0	0
7	1	1
8	0	0
9	1	0
10	0	1

En los problemas de clasificación binaria, existen cuatro posibles casos que se pueden presentar al comparar el resultado esperado con el resultado estimado por el modelo:

True Positive (TP) o Verdadero Positivo

- ✓ Ocurre cuando el resultado esperado es 1 y el modelo predice correctamente un 1. Esto indica que el modelo identificó correctamente una instancia positiva.

True Negative (TN) o Verdadero Negativo

- ✓ Ocurre cuando el resultado esperado es 0 y el modelo predice correctamente un 0. Esto indica que el modelo identificó correctamente una instancia negativa.

False Positive (FP) o Falso Positivo

- ✓ Ocurre cuando el resultado esperado es 0, pero el modelo predice un 1. Esto representa un error en el que el modelo clasifica incorrectamente una instancia negativa como positiva. Este tipo de error también se conoce como un "falso alarma".

False Negative (FN) o Falso Negativo

- ✓ Ocurre cuando el resultado esperado es 1, pero el modelo predice un 0. Esto representa un error en el que el modelo clasifica incorrectamente una instancia positiva como negativa. Este tipo de error puede ser crítico en problemas donde las instancias positivas son de alta importancia, como en la detección de enfermedades.

Si se aplica esto a la tabla anterior, tendríamos lo siguiente:

Número de muestra	Resultado esperado	Resultado estimado	Tipo de caso
1	1	1	TP
2	0	0	TN
3	1	0	FN
4	0	1	FP
5	1	1	TP
6	0	0	TN
7	1	1	TP
8	0	0	TN
9	1	0	FN
10	0	1	FP

La matriz de confusión tiene la siguiente forma general:

	Predicción Positiva	**Predicción Negativa**
Clase Positiva	True Positive (TP)	False Negative (FN)
Clase Negativa	False Positive (FP)	True Negative (TN)

Esto, aplicado al ejemplo anterior, quedaría de la siguiente manera:

	Predicción Positiva	Predicción Negativa
Clase Positiva	TP = 3	FN = 2
Clase Negativa	FP = 2	TN = 3

La matriz de confusión proporciona una visualización clara del desempeño del modelo de clasificación, destacando las predicciones correctas (true positives y true negatives) y los errores (false positives y false negatives).

A partir de esta matriz, se pueden calcular métricas como accuracy, recall, F1-score y precision para evaluar la efectividad del modelo.

Métricas derivadas de la matriz de confusión

Accuracy

El modelo matemático del **accuracy** es:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Un valor alto de *accuracy* indica que el modelo tiene un buen desempeño general, aunque puede no ser suficiente en casos de **desbalance de clases**.

Métricas derivadas de la matriz de confusión

Recall

El **recall**, también conocido como *sensibilidad* o *exhaustividad*, mide la capacidad del modelo para identificar correctamente las instancias positivas. Es especialmente útil en problemas donde las instancias positivas son críticas.

$$\text{Recall} = \frac{TP}{TP + FN}$$

Un valor alto de *recall* indica que el modelo tiene una buena capacidad para detectar las instancias positivas, aunque puede no reflejar el desempeño general si hay muchos falsos positivos.

Métricas derivadas de la matriz de confusión

Precisión (Precision)

La **precisión** mide la proporción de instancias positivas correctamente clasificadas en relación con todas las instancias clasificadas como positivas.

Es útil para evaluar la calidad de las predicciones positivas realizadas por el modelo.

$$\text{Precision} = \frac{TP}{TP + FP}$$

Un valor alto de *precisión* indica que el modelo tiene una buena capacidad para evitar **falsos positivos**, aunque puede no reflejar el desempeño general si hay muchos **falsos negativos**.

Métricas derivadas de la matriz de confusión

F1-score

El **F1-score** es la *media armónica* entre la **precisión** y el **recall**.

Es una métrica útil cuando se busca un equilibrio entre ambos, especialmente en problemas con **desbalance de clases**.

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Un valor alto de *F1-score* indica que el modelo tiene un buen **equilibrio** entre precisión y recall, y es adecuado para evaluar el desempeño cuando ambas métricas son importantes.

Métricas derivadas de la matriz de confusión

Especificidad (Specificity)

La **especificidad** mide la capacidad del modelo para identificar correctamente las instancias **negativas**.

Es útil en problemas donde es importante minimizar los **falsos positivos**.

$$\text{Specificity} = \frac{TN}{TN + FP}$$

Un valor alto de *especificidad* indica que el modelo tiene una buena capacidad para clasificar correctamente las instancias negativas, aunque puede no reflejar el desempeño general si hay muchos **falsos negativos**.

Métricas derivadas de la matriz de confusión

Sensibilidad (Sensitivity / Recall)

La **sensibilidad** mide la capacidad del modelo para identificar correctamente las instancias **positivas**.

Es especialmente útil en problemas donde las instancias positivas son de alta importancia.

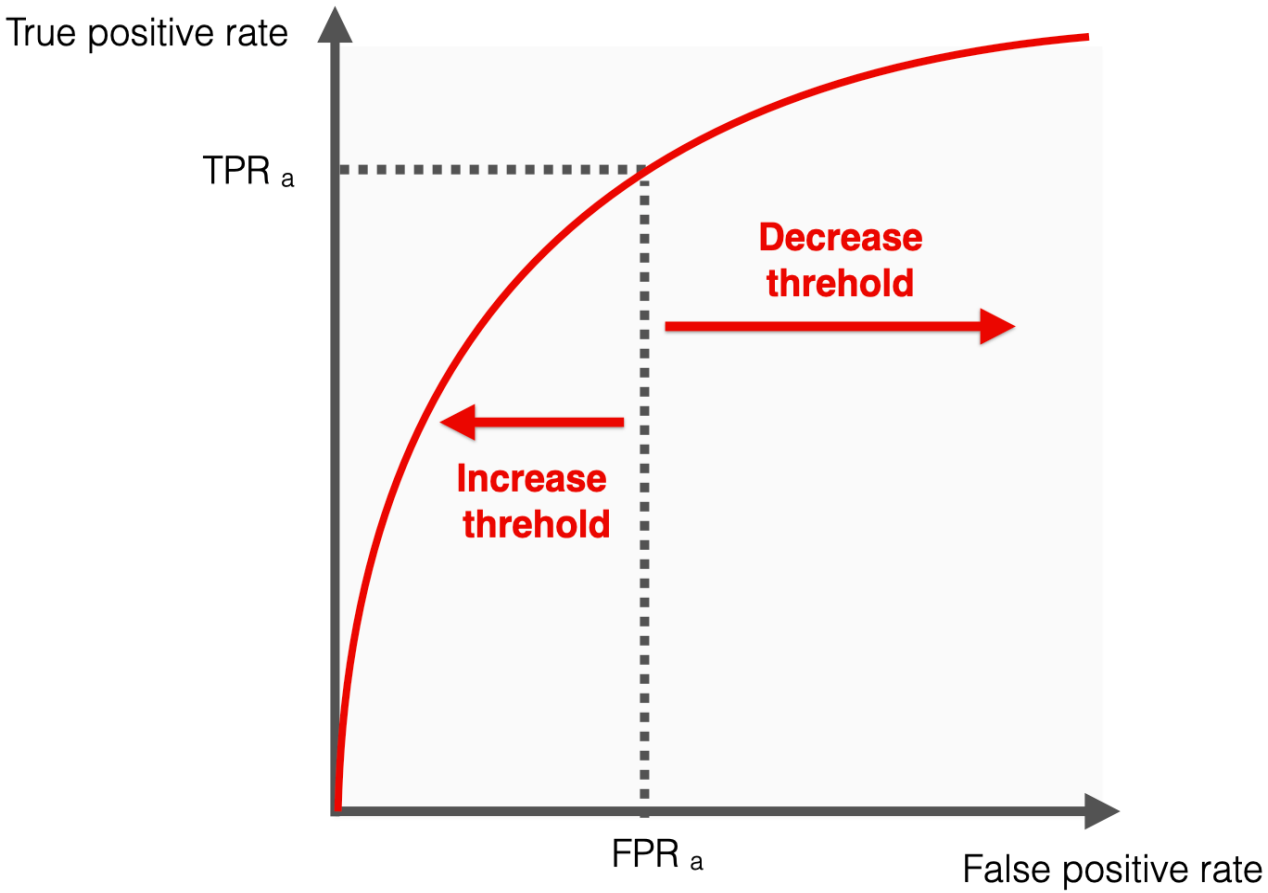
$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

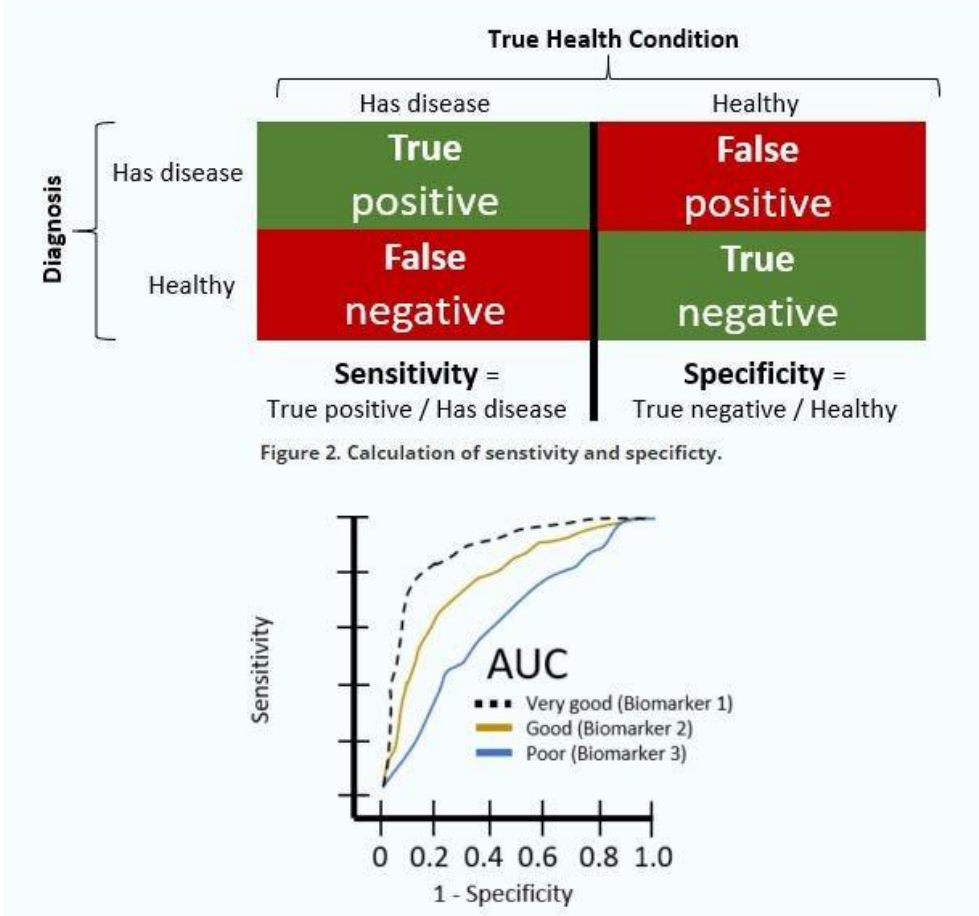
Un valor alto de *sensibilidad* indica que el modelo tiene una buena capacidad para detectar las instancias positivas, aunque puede no reflejar el desempeño general si hay muchos **falsos positivos**.

ROC Curve

- La curva ROC (Receiver Operating Characteristic) es una representación gráfica que muestra la relación entre la tasa de verdaderos positivos (TPR o sensibilidad) y la tasa de falsos positivos (FPR) para diferentes umbrales de clasificación. Es una herramienta útil para evaluar el desempeño de modelos de clasificación, especialmente en problemas con desbalance de clases.
- El área bajo la curva ROC (AUC-ROC) es una métrica derivada que mide la capacidad del modelo para distinguir entre clases. Un valor de AUC cercano a 1 indica un buen desempeño del modelo, mientras que un valor cercano a 0.5 indica un desempeño similar al azar.
- La curva ROC se utiliza para comparar modelos y seleccionar el umbral óptimo que maximice la sensibilidad y especificidad según el contexto del problema.

ROC Curve


True positive rate
TPR a
Decrease threhold
Increase threhold
FPR a
False positive rate


True Health Condition
Has disease
Healthy
Diagnosis
Has disease
Healthy
True positive
False positive
False negative
True negative
Sensitivity = True positive / Has disease
Specificity = True negative / Healthy
Figure 2. Calculation of senstivity and specificty.
Sensitivity
AUC
Very good (Biomarker 1)
Good (Biomarker 2)
Poor (Biomarker 3)
0 0.2 0.4 0.6 0.8 1.0
1 - Specificity

Ejemplo de ROC

AUC calculation

Acceder al ejemplo

El problema del imbalance de clases

- Cuando las clases están desbalanceadas, es decir, una clase tiene significativamente más ejemplos que otra, evaluar el desempeño de un modelo puede ser complicado.
- Métricas como el accuracy pueden ser engañosas en estos casos, ya que un modelo que predice siempre la clase mayoritaria puede tener un accuracy alto, pero un desempeño pobre en términos de identificar correctamente la clase minoritaria.
- Para abordar este problema, es importante considerar métricas adicionales que proporcionen una evaluación más completa del modelo.

Por ejemplo, supongamos que tenemos un dataset con 100 muestras con un imbalance de clases de 9:1, donde 90 muestras pertenecen a la clase negativa (0) y 10 muestras a la clase positiva (1).

Si decidimos ignorar la clase positiva y asumimos que todas las muestras pertenecen a la clase negativa, obtendríamos la siguiente matriz de confusión:

	Predicción Negativa	Predicción Positiva
Clase Negativa	TN = 90	FP = 0
Clase Positiva	FN = 10	TP = 0

En este caso, el modelo tendría un **accuracy** de:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{0 + 90}{0 + 90 + 0 + 10} = 0.9$$

Aunque el accuracy es alto (90%), el modelo no identifica correctamente ninguna instancia de la clase positiva, lo que lo hace inútil para problemas donde la clase positiva es crítica.

Sin embargo, cuando evaluamos las otras métricas, obtenemos lo siguiente:

1. **Recall (Sensibilidad)**

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{0}{0 + 10} = 0$$

El modelo no identifica ninguna instancia positiva.

2. **Precision**

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{0}{0 + 0} = \text{Indefinido}$$

No hay predicciones positivas, por lo tanto la precisión no se puede calcular.

3. **F1-score**

   El F1-score es la media armónica entre precisión y recall.

   Dado que el recall es 0 y la precisión es indefinida, el F1-score también es 0.

   Esto indica un desempeño muy pobre en la detección de la clase positiva.

Pero, ¿qué tal si tenemos un modelo que en el mismo dataset predice correctamente 9 de las 10 clases positivas y 81 de las 90 clases negativas?

Esto nos daría la siguiente matriz de confusión:

	Predicción Negativa	Predicción Positiva
Clase Negativa	TN = 81	FP = 9
Clase Positiva	FN = 1	TP = 9

A partir de esta matriz de confusión, podemos calcular las siguientes métricas

1. **Accuracy**

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{9 + 81}{9 + 81 + 9 + 1} = \frac{90}{100} = 90\%$$

Aunque el *accuracy* es igual al caso anterior, este modelo **mejora sustancialmente** en la detección de positivos.

2. **Precision**

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{9}{9 + 9} = \frac{9}{18} = 50\%$$

El modelo tiene una precisión **moderada** para identificar correctamente las clases positivas.

3. **Recall (Sensibilidad)**

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{9}{9 + 1} = \frac{9}{10} = 90\%$$

Alta capacidad para detectar correctamente las clases positivas.

4. **F1-score**

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = 2 \cdot \frac{0.5 \cdot 0.9}{0.5 + 0.9} = \frac{0.9}{1.4} \approx 64.29\%$$

Otro aspecto importante a tomar en cuenta a la hora de evaluar un modelo es el **decision threshold**, el cual puede impactar directamente las métricas del clasificador.

El **decision threshold** es el valor que determina el punto de corte para clasificar una instancia como positiva o negativa.

Por defecto, muchos clasificadores utilizan un threshold de 0.5, lo que significa que si la probabilidad estimada de una instancia pertenece a la clase positiva es mayor o igual a 0.5, se clasifica como positiva; de lo contrario, se clasifica como negativa.

El **decision threshold** puede hacer que un modelo sea más o menos sensible a una clase específica, lo cual tiene un impacto directo en la matriz de confusión y en las métricas derivadas. Ajustar el threshold permite controlar cómo el modelo clasifica las instancias, afectando la proporción de verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos.

Moviendo el threshold hacia valores más altos, el modelo se vuelve más conservador, clasificando menos instancias como positivas.

Esto puede aumentar la precisión al reducir los falsos positivos, pero también puede disminuir el recall al aumentar los falsos negativos.

Por otro lado, moviendo el threshold hacia valores más bajos, el modelo se vuelve más permisivo, clasificando más instancias como positivas. Esto puede aumentar el recall al reducir los falsos negativos, pero también puede disminuir la precisión al aumentar los falsos positivos.

Ejemplo de decision threshold evolution

Acceder al ejemplo

Una manera efectiva de evaluar los clasificadores binarios es con el **ROC AUC, la cual es** es una métrica efectiva para evaluar modelos de clasificación binaria.

- ✓ **Independencia del decision threshold:** Evalúa el desempeño del modelo en todos los posibles valores de threshold, proporcionando una visión completa de su capacidad para distinguir entre clases.
- ✓ **Robustez frente al class imbalance:** No se ve afectado por el desbalance de clases, ya que analiza la relación entre la tasa de verdaderos positivos (TPR) y la tasa de falsos positivos (FPR).
- ✓ Interpretación intuitiva
  - ROC AUC cercano a 1 indica excelente desempeño.
  - ROC AUC de 0.5 indica desempeño similar al azar.
  - ROC AUC menor a 0.5 indica clasificación incorrecta.
- ✓ **Visualización clara:** La curva ROC permite identificar el threshold óptimo para maximizar sensibilidad y especificidad.
- ✓ **Aplicaciones prácticas:** Útil en problemas médicos y financieros para ajustar el threshold según los objetivos del proyecto.