Inteligencia Artificial
Apuntes Semana 5, Clase #2

Luis Fernando Benavides Villegas
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica
lubenavides@estudiantec.cr

***Abstract*—Este documento recopila los apuntes de la clase del jueves 04 de septiembre de 2025 para el curso de Inteligencia Artificial. Se repasan conceptos clave de regresión lineal y sus limitaciones, así como los problemas de *overfitting* y *underfitting*. También se describen técnicas de subdivisión de datasets y estrategias para mejorar la capacidad de generalización de los modelos. Finalmente, se introduce la regresión logística como un modelo de clasificación binaria, explicando la función sigmoide, su derivada y el proceso de optimización de parámetros mediante descenso del gradiente.**

***Index Terms*—Inteligencia Artificial, regresión lineal, regresión logística, función sigmoide, overfitting, underfitting, optimización**

I. Noticias de la Semana

A. IngenIEEEría Costa Rica

Un evento de Ingeniería que organiza IEEE Costa Rica. Habrá charlas de profesores distinguidos en diversas áreas y participación de empresas. [1]

B. Referencias falsas en IA

Las "alucinaciones" en inteligencia artificial son cuando los modelos generan referencias aparentemente válidas pero que en realidad no existen. Esto fue debatido en el grupo Parma del TEC y se resaltó la importancia de siempre verificar las fuentes. La responsabilidad recae en el usuario de confirmar la veracidad de la información antes de tomarla como cierta.

C. Google Nano Banana

El nuevo modelo de Google enfocado en la edición de imágenes que se llama *Nano Banana*. A diferencia de otros generadores que recrean la imagen completa desde cero, este modelo conserva mejor los detalles originales y el contexto. Así, al editar una foto mantiene la coherencia entre iteraciones. Se habló también de posibles sesgos en los modelos, al notar que repeticiones en imágenes de personas modificaban rasgos hacia un perfil más latino.

II. Repaso de la Clase Anterior

A. Potenciales problemas al aplicar una Regresión Lineal

1) No linealidad: Un supuesto de la regresión lineal es que la relación entre las variables predictoras y la variable respuesta es lineal. Cuando no se cumple, los residuos muestran patrones sistemáticos (por ejemplo, en forma de parábola) en lugar de distribuirse de manera aleatoria. Esto indica que el modelo lineal no es adecuado. Para resolverlo, una opción es aplicar **feature engineering**, agregando términos polinómicos que transformen las variables originales y permitan que la relación se aproxime mejor a una forma lineal. De esta manera, aunque la relación real sea curva, el modelo puede ajustarse con menor error.

2) Datos sobresalientes: Surgen por ruido, errores de medición o datos atípicos y pueden afectar el ajuste del modelo. Una forma de tratarlos es **estandarizar los residuos** dividiendo entre la desviación estándar. Una vez normalizados, se mide cuántas desviaciones estándar se aleja cada dato. Si un dato está muy lejos (más de 2 o 3 desviaciones estándar), se considera sobresaliente. Otras técnicas que vimos fueron el **rango intercuartílico**, que es el que se usa en gráficos de caja y bigotes, y la **Winsorización**, donde en vez de eliminar datos atípicos se reemplazan por valores en percentiles límite.

3) Colinealidad: Se da cuando dos o más predictores están altamente correlacionados entre sí. Esto hace difícil separar el efecto de cada variable en la predicción, afectando la estabilidad de los coeficientes del modelo. En consecuencia, los parámetros estimados se vuelven poco confiables y muy sensibles a cambios en los datos. Para detectarla, se pueden usar medidas como el **VIF (Variance Inflation Factor)**. Una solución común es eliminar variables redundantes o aplicar técnicas de regularización.

B. Dataset

Es el conjunto completo de datos disponibles para entrenar y evaluar un modelo. Normalmente se subdivide en diferentes partes para poder medir la capacidad de generalización y evitar problemas como el *overfitting*.

C. Training Set

Subconjunto usado para entrenar el modelo y ajustar sus parámetros. Es donde el algoritmo aprende los patrones presentes en los datos.

D. Validation Set

Subconjunto usado durante el entrenamiento para evaluar el rendimiento intermedio del modelo. Sirve para medir si

lo aprendido se generaliza a datos no vistos y para ajustar hiperparámetros. Permite detectar problemas de sobreajuste de manera temprana sin necesidad de esperar a la prueba final.

E. Técnicas para subdividir el dataset

1) Random Sampling: Consiste en dividir aleatoriamente los datos entre entrenamiento y prueba. Es adecuado cuando las clases están balanceadas, ya que garantiza representatividad sin introducir sesgos. El problema surge si las clases están desbalanceadas, porque puede que un subconjunto quede con muy pocos o incluso sin ejemplos de alguna clase.

2) Stratified Sampling: Se usa cuando las clases están desbalanceadas. Mantiene la misma proporción de clases en los conjuntos de entrenamiento y prueba. De esta forma, si en el dataset original una clase representa el 90% y otra el 10%, esa relación se conserva en las divisiones.

3) K-Fold Cross-Validation: El conjunto de entrenamiento se divide en K partes (folds). En cada iteración se usa $K-1$ folds para entrenar y el fold restante para validar. El proceso se repite K veces, rotando el fold de validación. Esto permite aprovechar mejor los datos disponibles y obtener una evaluación más robusta del modelo.

F. Posibles escenarios de comportamiento de training y validation

1) Overfitting: El error en training es bajo pero el error en validation comienza a aumentar después de cierto punto. El modelo memoriza los datos de entrenamiento en lugar de aprender patrones generales. Se captura también el ruido de los datos, lo que provoca que no pueda generalizar. Se caracteriza por tener **alta varianza**.

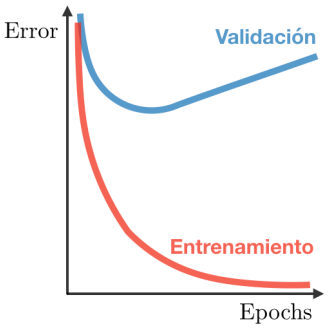

Fig. 1. Ej. de overfitting

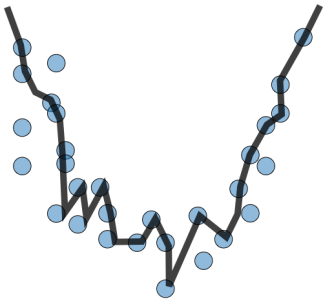

Fig. 2. Ej. de regresión de overfitting

Una técnica para evitarlo es el *early stopping*, que consiste en detener el entrenamiento en la época donde el error de validación empieza a empeorar.

2) Underfitting: Tanto el error en training como en validation son altos. El modelo no logra aprender patrones de los datos porque es demasiado simple o incorrecto para el problema. Se caracteriza por **alto sesgo**, es decir, asume una forma equivocada de los datos (por ejemplo, usar un modelo lineal para datos con comportamiento cuadrático).

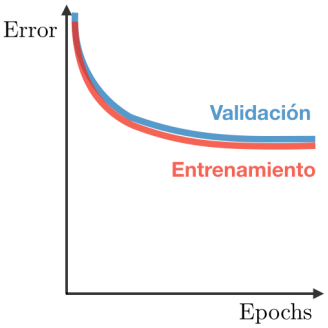

Fig. 3. Ej. de underfitting

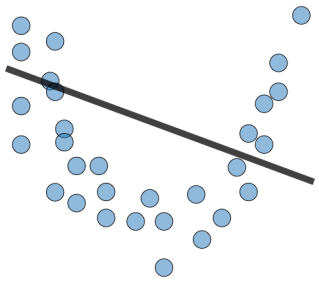

Fig. 4. Ej. de regresión de underfitting

3) Caso ideal: El error en training es bajo y también lo es en validation. El modelo logra ajustarse a los datos sin sobreajustarse al ruido y puede generalizar bien a ejemplos no vistos. Representa un buen equilibrio entre sesgo y varianza.

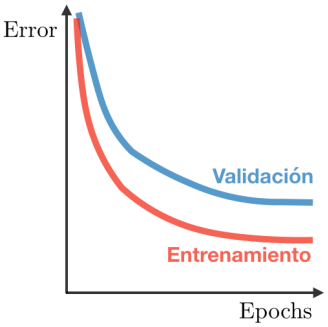

Fig. 5. Ej. del caso ideal

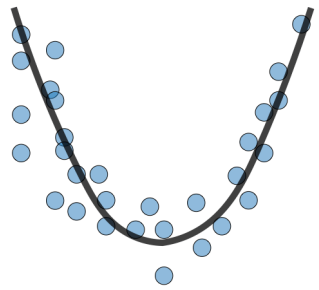

Fig. 6. Ej. de regresión del caso ideal

4) Bias-Variance Tradeoff: Es un caso súper raro porque el error en training es alto pero el error en validation es bajo. Si sucede, puede deberse a errores de cálculo o valores mal tomados, no a un aprendizaje real del modelo.

III. Alto Bias

Se presenta cuando el modelo es demasiado simple y no logra capturar el patrón real de los datos, provocando **underfitting**. Tanto el error en training como en validation son altos, ya que el modelo asume demasiado sobre la forma de los datos.

A. Causas

- Modelo demasiado simple (ej. lineal para datos con relaciones cuadráticas).
- No se utilizan todas las variables relevantes.
- Los *features* disponibles no son buenos predictores de la variable objetivo.

B. Posibles soluciones

- Incrementar la complejidad del modelo (por ejemplo, pasar de lineal a cuadrático o a un modelo más flexible).
- Incorporar más *features* o transformar los existentes.

- Sustituir o recolectar mejores *features* que representen de manera adecuada el problema.

IV. Alta Varianza

Se presenta cuando el modelo se ajusta demasiado a los datos de entrenamiento pero falla al generalizar en el conjunto de validación. Esto provoca **overfitting**, donde pequeñas variaciones en los datos de entrada pueden generar malas predicciones.

A. Causas

- El modelo es demasiado complejo y aprende patrones irrelevantes o ruido.
- Exceso de dimensionalidad: agregar muchas variables aumenta el riesgo de overfitting.
- Muy pocos ejemplos en el conjunto de entrenamiento, especialmente en problemas con clases desbalanceadas.

B. Posibles soluciones

- Reducir la complejidad del modelo (ej. usar menos capas o un modelo más simple).
- Disminuir la dimensionalidad eliminando variables irrelevantes.
- Obtener más ejemplos de entrenamiento para mejorar la representación de todas las clases.
- Aplicar técnicas de regularización que penalizan la complejidad del modelo, como:
  - L1 y L2 (penalización sobre los parámetros).
  - Dropout (apagar ciertas neuronas durante el entrenamiento).

V. Regresión Logística

Aunque su nombre incluya "regresión", la regresión logística es un modelo de **clasificación**, no de regresión. Se utiliza principalmente para problemas **binarios**, donde las etiquetas y toman los valores 0 o 1.

A. Diferencia con la regresión lineal

- En **regresión lineal** se predicen valores continuos en los reales ($\mathbb{R}$).
- En **regresión logística** se predice la probabilidad de pertenecer a una clase u otra. El resultado final es una clasificación: 0 o 1.

Por ejemplo, con una variable como el tamaño de una calabaza:

- Regresión lineal: predice el precio aproximado en valores reales.
- Regresión logística: predice si la calabaza es naranja (1) o no lo es (0).

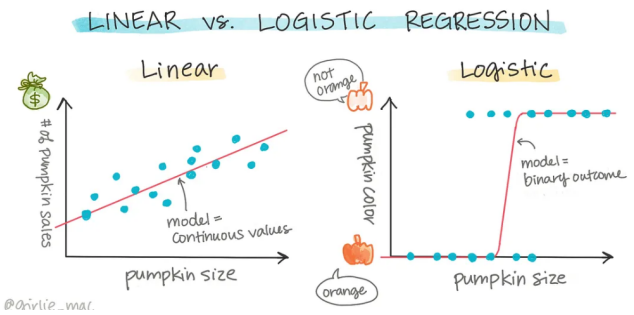

Fig. 7. Regresión lineal vs logística

B. Distribución de Bernoulli

Cada etiqueta y_i es una variable aleatoria que sigue una distribución de Bernoulli. La probabilidad de que ocurra el evento ($y = 1$) o no ocurra ($y = 0$) se define como:

$$P(Y = k) = p^k(1-p)^{1-k}, \quad k \in \{0, 1\}$$

donde:

- p es la probabilidad de éxito ($y = 1$).
- k es la etiqueta observada (0 o 1).

Así, si $k = 1$, la probabilidad es p; y si $k = 0$, la probabilidad es $1 - p$.

VI. Función Sigmoide

La función sigmoide es una herramienta clave porque introduce **no linealidad** al modelo y tiene un **rango de salida entre 0 y 1**, lo cual la hace ideal para trabajar con probabilidades.

Se define como:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Al tomar valores de entrada muy negativos, la salida se acerca a 0; mientras que con valores grandes y positivos, se acerca a 1.

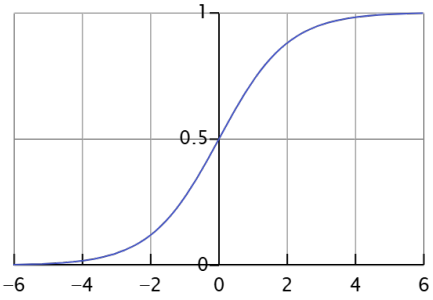

Fig. 8. Gráfica de la función sigmoide

Además, el argumento x puede ser cualquier valor o incluso otra función (composición de funciones), lo que da flexibilidad para modelar relaciones más complejas.

La idea es tomar la salida de un modelo lineal y convertirla en una probabilidad. Si partimos de una función lineal $f_{w,b}(x) = wx+b$, al aplicarle la función sigmoide obtenemos:

$$\hat{y} = \sigma(f_{w,b}(x)) = \frac{1}{1+e^{-(wx+b)}}$$

De esta forma:

- Si $\hat{y} < 0.5$, se clasifica como 0.
- Si $\hat{y} \geq 0.5$, se clasifica como 1.

Esto convierte la regresión logística en un modelo de **clasificación binaria**. Queremos hacerlo así porque calcular una función lineal es simple computacionalmente, es un buen método para mantener la relación entre variables y pesos y permite modelar problemas con mayor complejidad.

A. Diagrama Computacional de la Regresión Logística

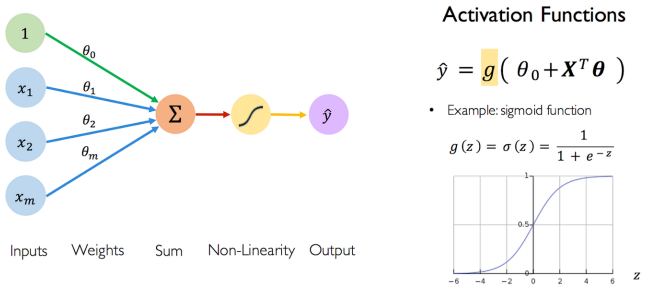

Fig. 9. Diagrama

1) Los **inputs** (features) x ingresan junto con un vector de **pesos** w y un **bias** b.
2) Se calcula el producto punto entre el vector x y el vector w.
3) Se le aplica la función no lineal $\sigma(z)$, obteniendo como salida una probabilidad.
4) Finalmente, esta probabilidad se compara con un umbral para asignar una etiqueta de clase (0 o 1).

En algunos textos, al valor lineal $z = wx + b$ se le llama **pre-activación**, y a la aplicación de la sigmoide se le llama **activación**. La salida de la activación corresponde a la probabilidad estimada $\hat{y}$.

B. Optimización

Nuestro objetivo es optimizar los parámetros w y b para que el modelo aprenda correctamente. Tenemos:

$$\hat{y} = \sigma(f_{w,b}(x)) = \frac{1}{1+e^{-(wx+b)}}$$

Para ajustar los parámetros, necesitamos calcular las derivadas parciales de la función de pérdida respecto a w y b. Sin embargo, antes de derivar, debemos definir una función de pérdida adecuada.

En regresión lineal usamos el **error cuadrático medio (MSE)**, pero en clasificación esto deja de ser útil, porque ya no predecimos valores continuos, sino **probabilidades**.

El procedimiento general sigue siendo el mismo:

- Definimos una función de pérdida L apropiada para probabilidades.
- Calculamos sus derivadas respecto a w y b.
- Usamos esas derivadas en el algoritmo de **descenso del gradiente**, iterando sobre los datos de entrenamiento para ir actualizando los parámetros y minimizar la pérdida.

C. Derivada de la función sigmoide

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

$$\sigma'(x) = \frac{1' \cdot (1+e^{-x}) - 1 \cdot (1+e^{-x})'}{(1+e^{-x})^2}$$

$$\sigma'(x) = \frac{e^{-x}}{(1+e^{-x})^2}$$

$$\sigma'(x) = \frac{e^{-x}+1-1}{(1+e^{-x})^2}$$

$$\sigma'(x) = \frac{e^{-x}+1}{(1+e^{-x})^2} - \frac{1}{(1+e^{-x})^2}$$

$$\sigma'(x) = \frac{1}{1+e^{-x}} - \frac{1}{(1+e^{-x})^2}$$

$$\sigma'(x) = \frac{1}{1+e^{-x}} \cdot \left(1 - \frac{1}{1+e^{-x}}\right)$$

$$\sigma'(x) = \sigma(x)\,(1-\sigma(x))$$

D. Hallar la función de pérdida

¿MSE? Esto y más en la siguiente clase.

VII. Conclusión

En esta clase se reforzaron conceptos esenciales para comprender cómo los modelos de aprendizaje supervisado aprenden a partir de datos. Se revisaron las limitaciones de la regresión lineal y los problemas comunes asociados al sesgo y la varianza, así como técnicas para evaluar y mejorar la generalización de los modelos. Además, se introdujo la regresión logística como un modelo de clasificación, destacando el papel de la función sigmoide y su derivada en el proceso de optimización. Estos fundamentos sientan la base para profundizar en funciones de pérdida específicas y en el entrenamiento de modelos más complejos en futuras sesiones.

REFERENCIAS

[1] IEEE Costa Rica. “IngenIEEEría Costa Rica.” [En línea]. Disponible: https://r9.ieee.org/costarica/ingenieeeria

[2] A. Shervine. “Hoja de referencia de Aprendizaje Automático.” Stanford University. [En línea]. Disponible: https://stanford.edu/~shervine/l/es/teaching/cs-229/hoja-referencia-aprendizaje-automatico-consejos-trucos