INTELIGENCIA ARTIFICIAL (1INFX18)

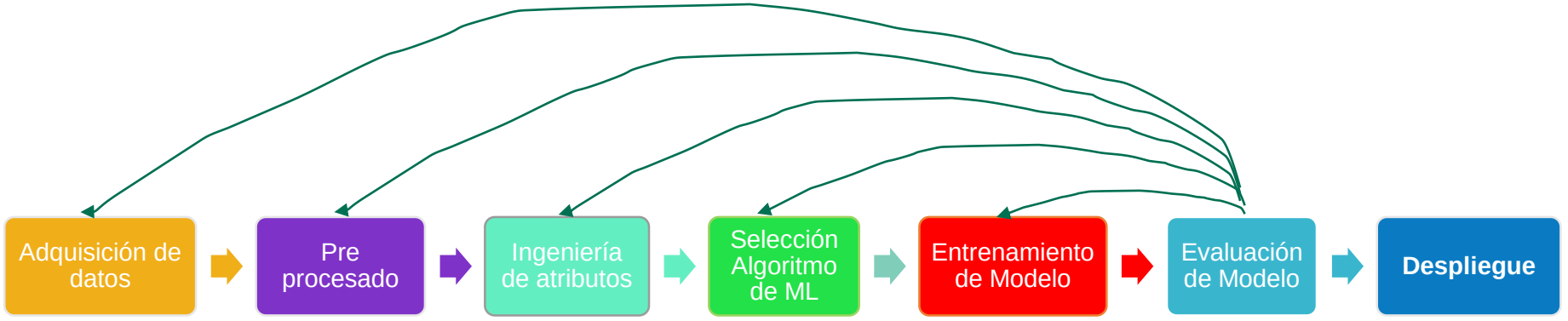
Unidad 2: Fundamentos de *Machine Learning* y redes neuronales artificiales

Tema 3: *Algoritmos básicos para regresión*

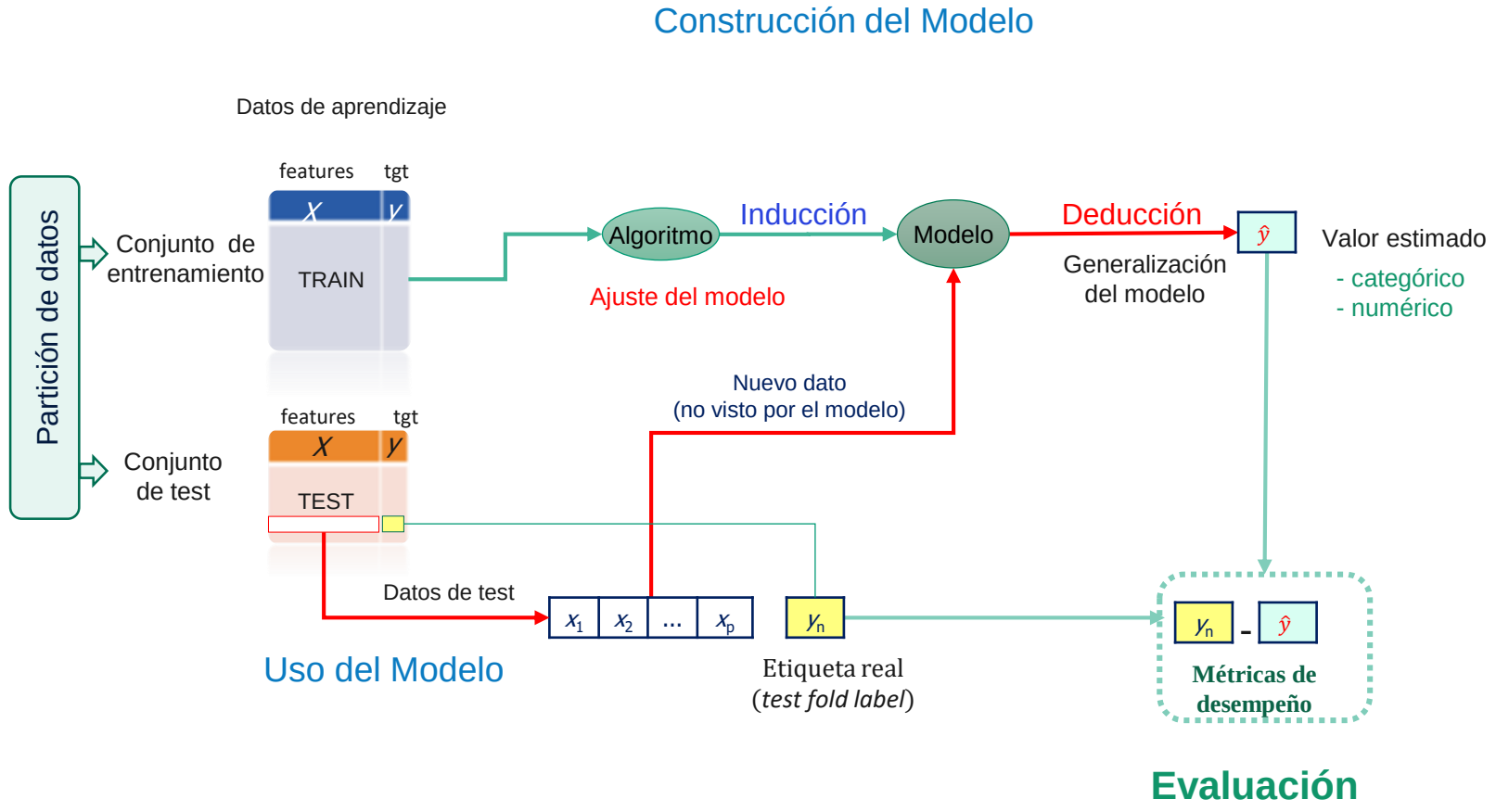
Dr. Edwin Villanueva Talavera

Contenido

- Partición de datos
- Regresión Lineal
- Regresión K-NN
- Arboles de Regresion

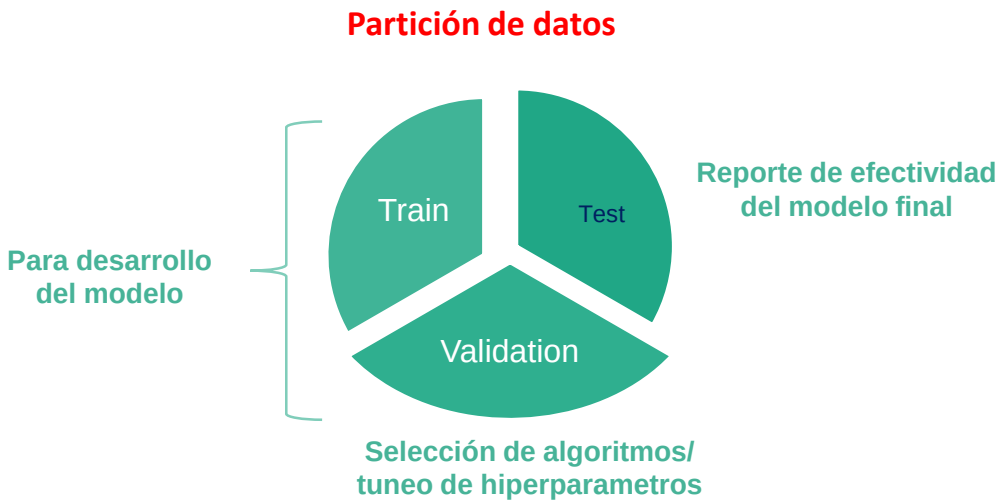


Proceso de Aprendizaje Supervisado



Partición de datos

- **TRAIN**: aprendizaje del modelo
- **VALIDATION**: testar/probar el modelo en su capacidad de generalización para hacer ajustes y mejoras
- **TEST**: para reportar la efectividad del modelo final

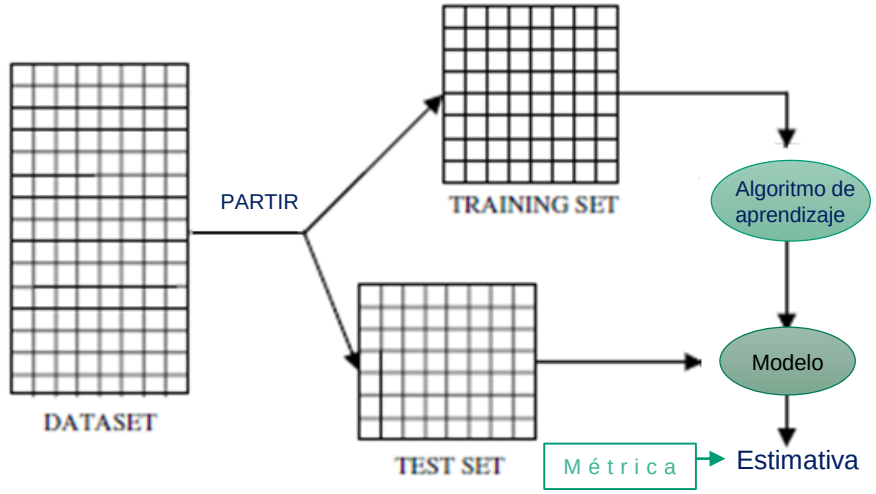


Partición de datos

- Método 1: Separar los datos en Train y Test

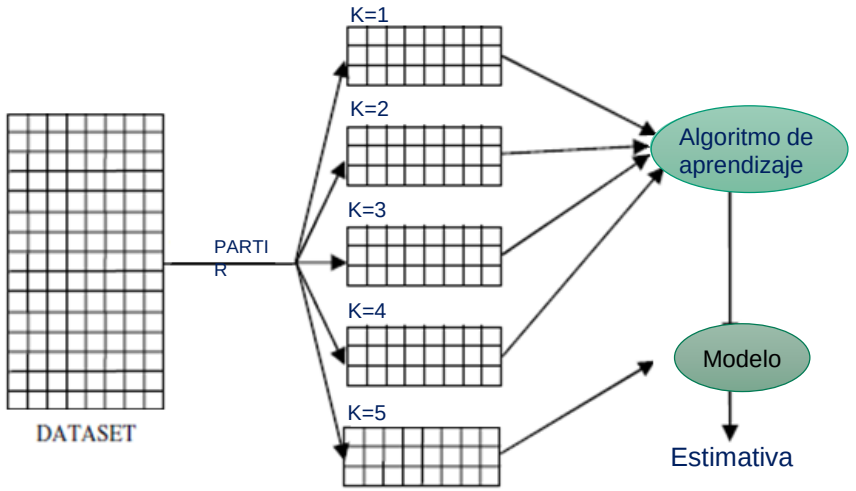
DATASET | TRAIN | TEST

Proporciones:
80:20, 70:30,
60:40, 67:33



Partición de datos

- Método 2: K-fold Cross Validation (Validación cruzada)



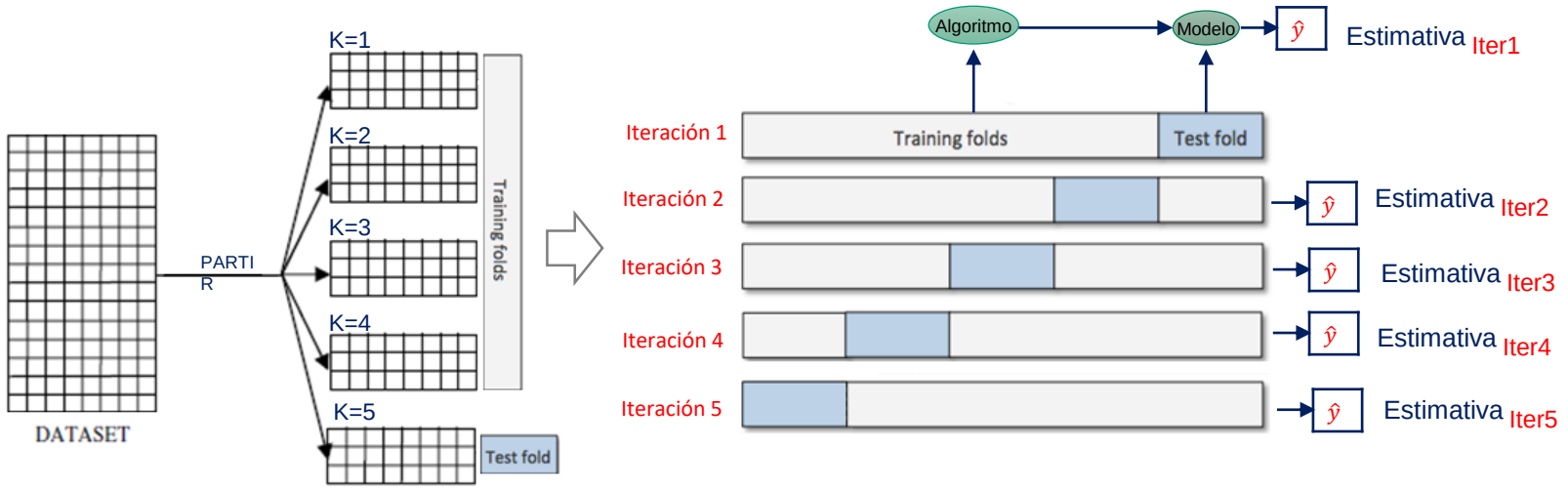
Ejemplo: $DATASET_{100 \times p}$, K =5 => Cada $fold_{1:5}$ = 20 registros.

$DATASET_{98 \times p}$, K =5 => Cada $fold_{1:4}$ = 20 registros

último $fold_5$ = 18 registros.

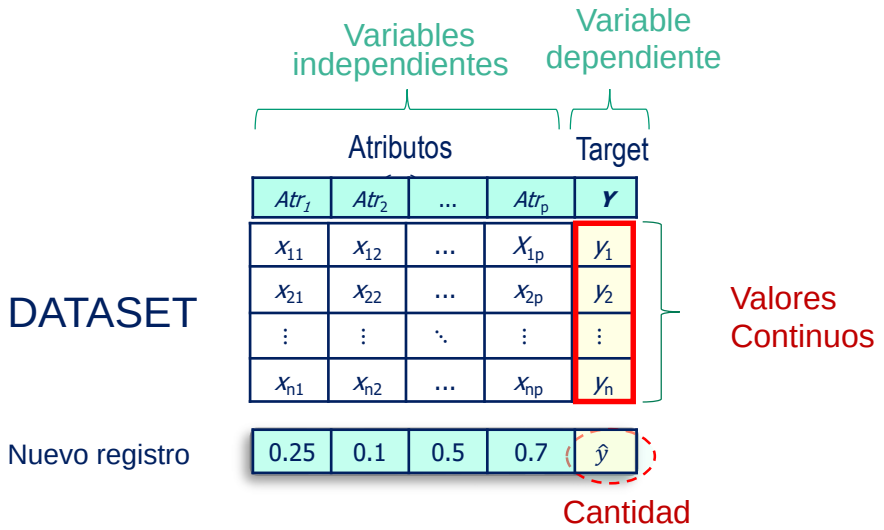
Partición de datos

- Método 2: K-fold Cross Validation (Validación cruzada)



Precisión Algoritmo = promedio($\text{Estimativa}_{\text{Iter1}}$, $\text{Estimativa}_{\text{Iter2}}$,…, $\text{Estimativa}_{\text{Iterk}}$)

El problema de Regresión



Tarea: aprender la relación de **x** con y :

$$\hat{y} = f(\boldsymbol{x})$$

Modelo

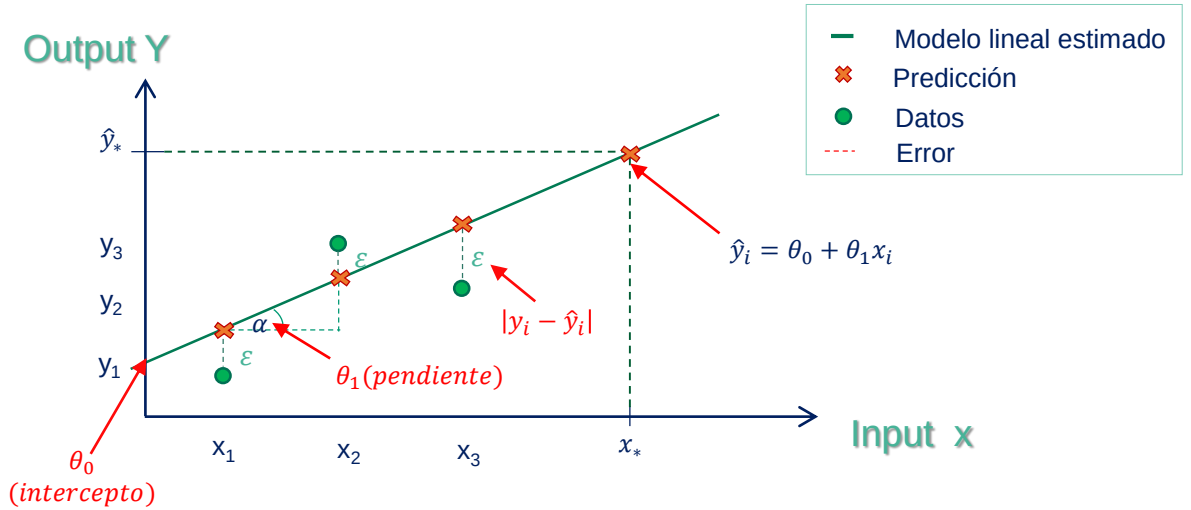
$$\hat{y} = f(\boldsymbol{x}) = y + \varepsilon$$

Error de predicción

Regresión lineal

Regresión Lineal Simple

$$\hat{y} = f(x) = \theta_0 + \theta_1 x$$



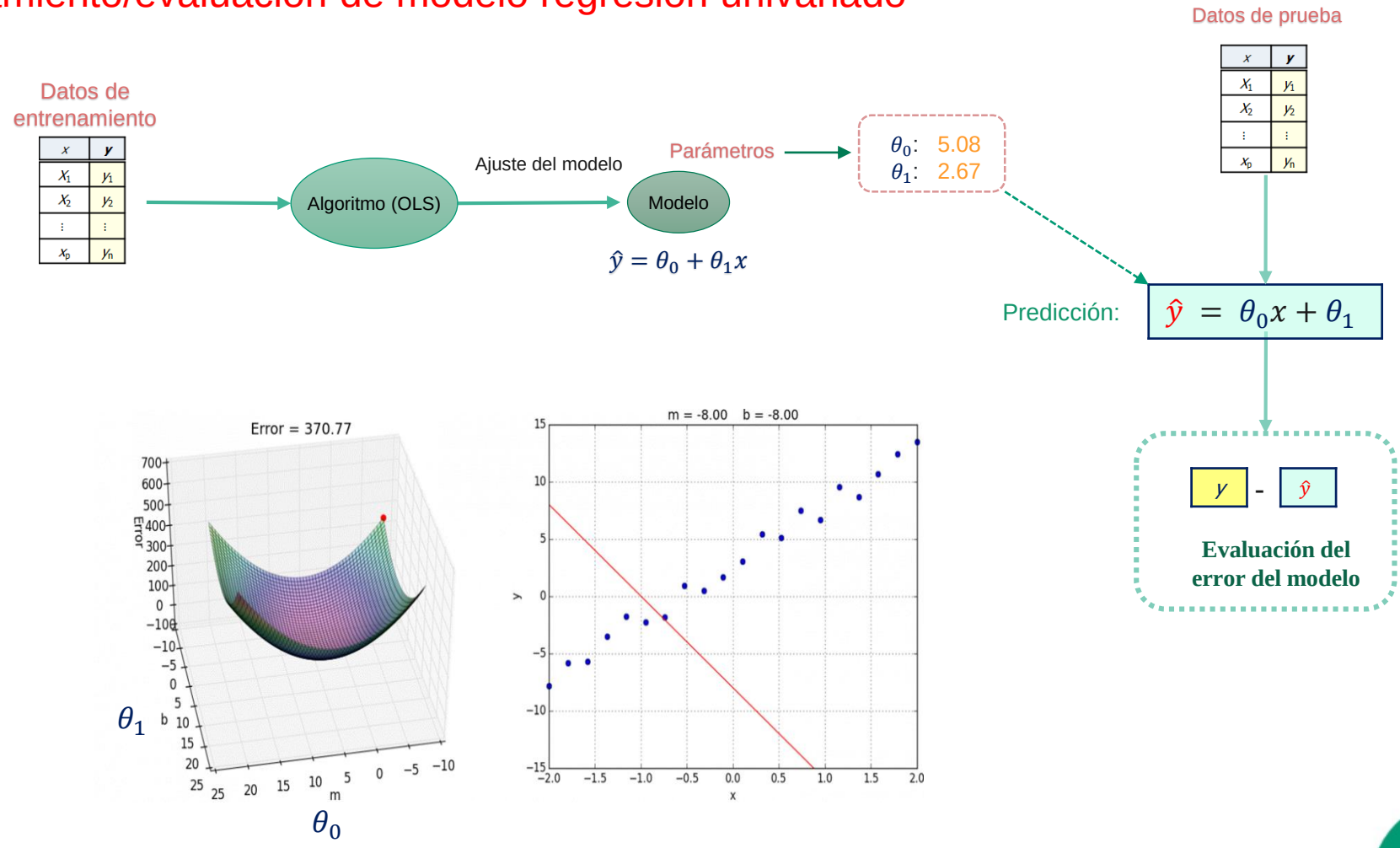
Función de Perdida: **Error cuadrático Medio:**

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

y_i es el valor real y $\hat{y}_i$ es la salida estimada para una entrada i.

Proceso de Aprendizaje Supervisado

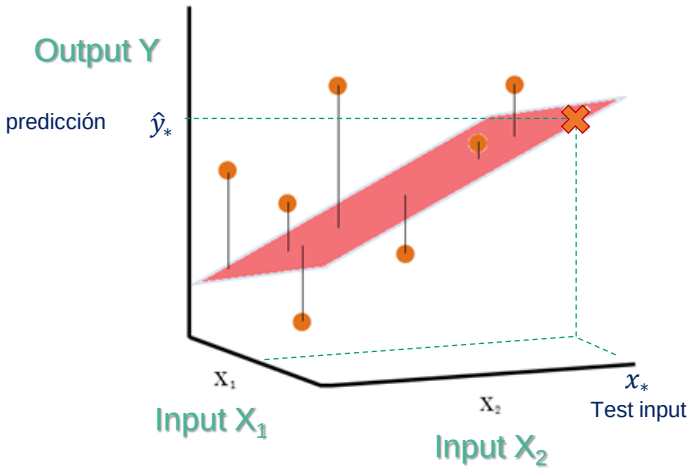
Ej. Entrenamiento/evaluación de modelo regresion univariado



Regresión lineal

Regresión Lineal Múltiple

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 \ldots + \theta_p x_p$$



Forma vectorial: $\hat{y} = \theta^T X$

Coeficientes

$$\boldsymbol{\theta}^T = [\theta_0, \theta_1, \theta_2, \ldots, \theta_p]$$

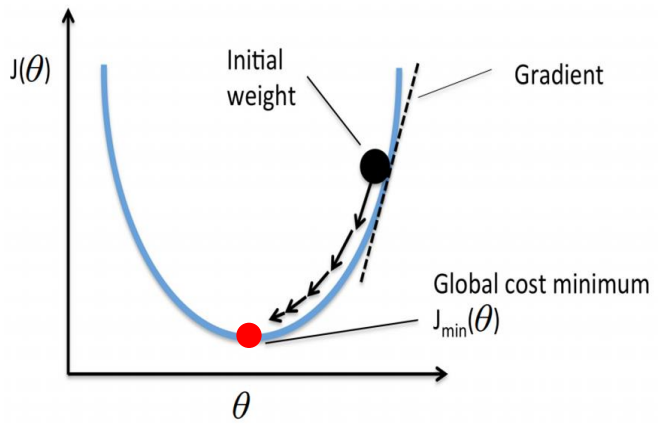
Atributos

$$\boldsymbol{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \ldots \end{bmatrix}$$

- *Ordinary least squares (OLS): Método de optimización de parámetros* $\boldsymbol{\theta}^T$

Reduce el error cuadrático medio de forma iterativa

$$MSE = J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

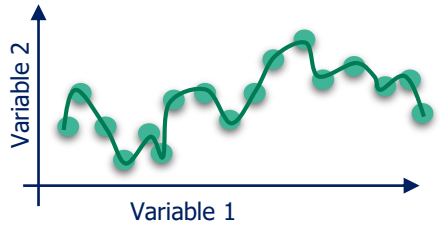


GRADIENTE DESCENDIENTE

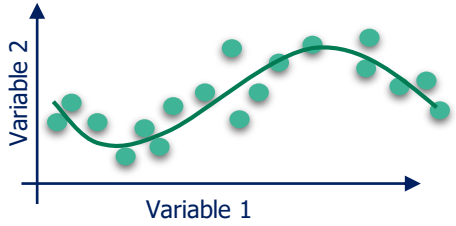
Overfitting y *underfitting* en Regresión

- *Overfitting (sobre ajuste)*

  Modelo sobre aprendido, no puede generalizar.
  El modelo falla sobre nuevos datos porque no tiene valores similares a las muestras de entrenamiento.

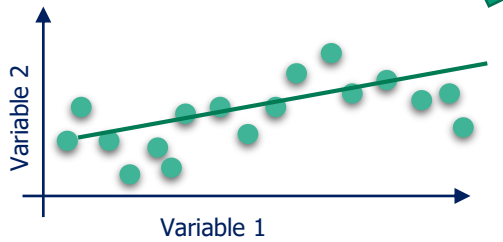


- Ajuste adecuado



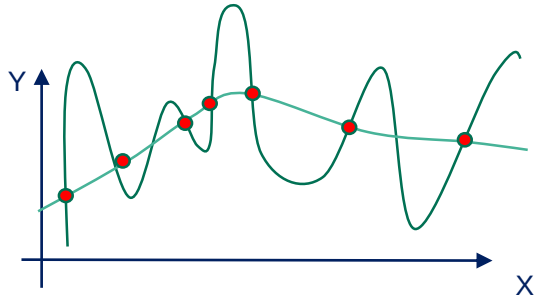
- *Underfitting (pobre ajuste)*

  Modelo con falta de aprendizaje, no puede generalizar
  Con nuevos datos, el modelo no genera la salida deseada por falta de suficientes ejemplos de aprendizaje.

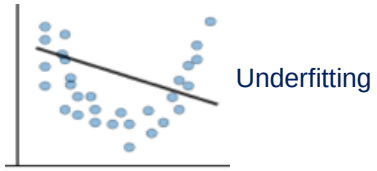


Regularización

- El procedimiento de regularización tiene como objetivo evitar que el modelo se ajuste demasiado a los datos (overfitting). Se reduce el tamaño de los coeficientes, manteniendo el mismo número de atributos.

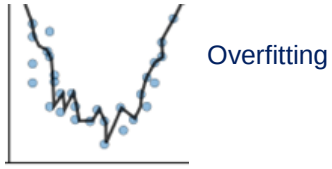


Modelo de menor capacidad



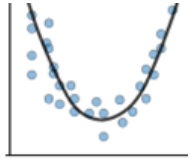
$\hat{y}_i = 3 - 0.5x_1$

Modelo sobreajustado



$\hat{y}_i = 0.4 + 0.2x_1 + 0.2x_2 + 0.7x_3 + \cdots$

Modelo regularizado



✓ Ajuste adecuado

$\hat{y}_i = 0.2x_1 + 0.6x_2$

Regularización

- *Ridge Regression*, es una extensión de la regresión lineal donde se modifica la función f(x) para minimizar la complejidad del modelo. Usa la norma L2 (distancia euclidiana) para minimizar la los valores de los parámetros θ^T.

Reduce los valores de los coeficientes $\boldsymbol{\theta^T}$.

$$J(\theta)_{Ridge} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \alpha \sum_{j=0}^{p}\theta_j^2, \qquad \theta^T = [\theta_0, \theta_1, \theta_2, \dots, \theta_p]$$

- LASSO (*Least Absolute Shrinkage and Selection Operator*), similar a Ridge *regression* los valores de los parámetros θ^T son minimizados usando la norma L1 (distancia de Hamming).

Elimina algunos coeficientes $\boldsymbol{\theta^T}$.

$$J(\theta)_{LASSO} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \alpha \sum_{j=0}^{p}|\theta_j|$$

- ElasticNet combina las propiedades de la regresión Ridge y LASSO. Busca minimizar la complejidad del modelo aprendido, penalizando el modelo usando tanto la norma L2 y la norma L1.

Reduce los valores de los coeficientes $\boldsymbol{\theta^T}$ y elimina alguno de ellos.

$$J(\theta)_{ElasticNet} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \alpha_1 \sum_{j=0}^{p}\theta_j^2 + \alpha_2 \sum_{j=0}^{p}|\theta_j|$$

K-Nearest Neighbors (KNN)

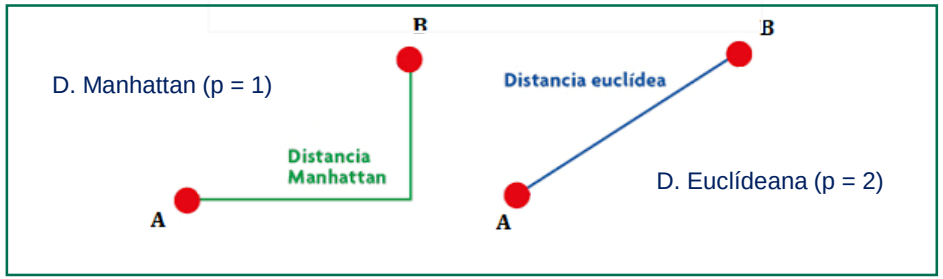
- KNN es un algoritmo usado en problemas de clasificación y regresión.
- Cuando ingresa un nuevo registro de datos, el algoritmo KNN ubica los k registros más similares en el conjunto de datos de entrenamiento. La variable de salida predicha puede ser la media o mediana de los k vecinos más cercanos.
- Se recomienda usar variables escaladas o estandarizadas.

```
from sklearn.neighbors import NearestNeighbors
neigh = NearestNeighbors(n_neighbors=5, metric='minkowski', p=2)
```

K

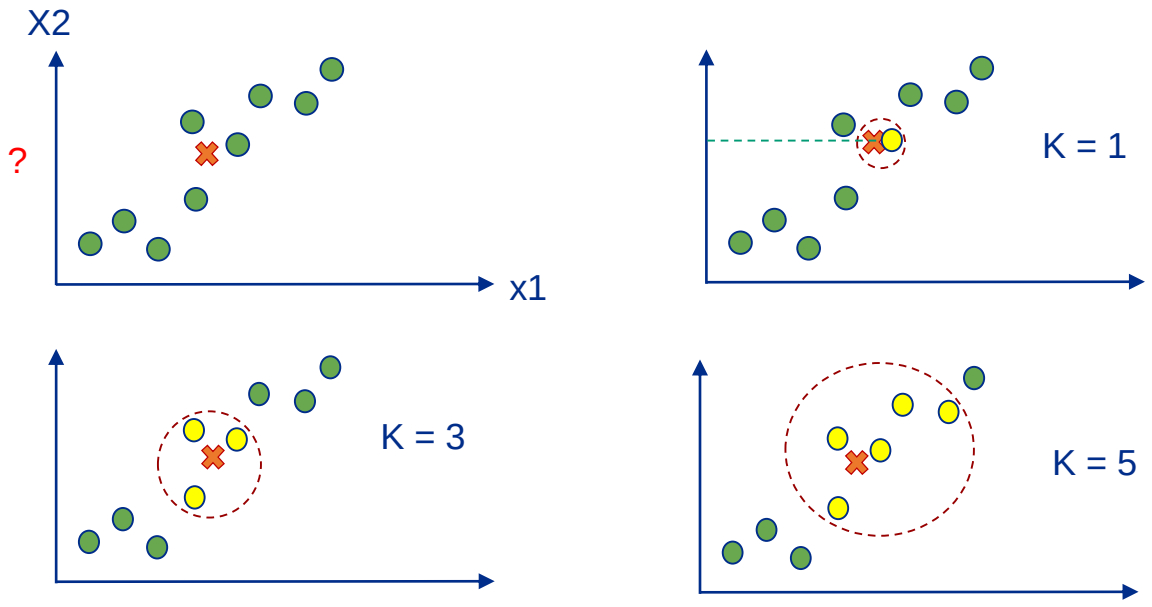
Distancia de Minkowski:

$$d(A,B) = \sqrt[p]{\sum_{i=1}^{n} (|A(x_i) - B(x_i)|)^p}, \qquad p \geq 1$$



K-Nearest Neighbors (KNN)

- Para determinar el valor predicho de salida, se consideran las medias de los k vecinos adyacentes más cercanos.



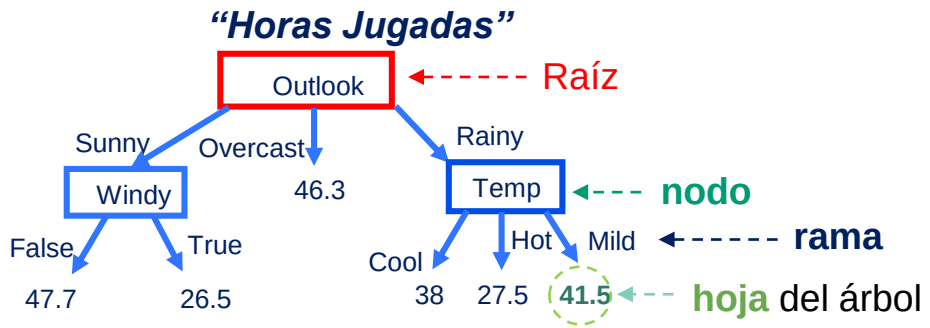

```
NearestNeighbors(n_neighbors=5, metric='minkowski', p =2, algorithm='auto')
```

Decision Trees

❑ CART (Classification And Regression Tree), es un método no paramétrico utilizado para clasificación y regresión:

- Nodo inicial es la **Raíz**
- **Nodos internos:** atributos del dataset.
- **Ramas:** son opciones de decisión (si, no), costo o probabilidad de que ocurra.
  - El conjunto de literales que etiquetan una rama se denomina partición.
- **Hojas:** representa una expresión lógica (posible resultado), que es la conjunción de los literales encontrados en la ruta desde la raíz del árbol hasta dicha hoja.
- Los valores de los nodos corresponden a la media de las observaciones en esa región.

Predictors				Target
Outlook	Temp	Humidity	Windy	Hours Played
Rainy	Hot	High	False	25
Rainy	Hot	High	True	30
Overcast	Hot	High	False	46
Sunny	Mild	High	False	45
Sunny	Cool	Normal	False	52
Sunny	Cool	Normal	True	23
Overcast	Cool	Normal	True	43
Rainy	Mild	High	False	35
Rainy	Cool	Normal	False	38
Sunny	Mild	Normal	False	48
Rainy	Mild	Normal	True	48
Overcast	Mild	High	True	52
Overcast	Hot	Normal	False	44
Sunny	Mild	High	True	30



Regresión con Decision Trees

Algoritmo de entrenamiento

1) Inicio: Comienza colocando todos los datos en un nodo (nodo raíz)

2) Evaluar divisiones: Para cada nodo hoja, el algoritmo evalúa cada posible división con la función de costo.

3) Escoge la mejor división: Elegir el nodo y la mejor división (la que minimiza la función de perdida) y crear 2 nodos hijos con dicha división

4) Repetir: Repetir pasos 2 y 3 hasta que se cumpla una condición de parada (se llega a un mínimo de datos en los nodos; profundidad máxima del árbol, etc.)

Función costo CART para regresión:

$$J(k, t_k) = \frac{m_{left}}{m} MSE_{left} + \frac{m_{right}}{m} MSE_{right}$$

Donde

$$\begin{cases} MSE_{node} = \sum_{i \in node} (\hat{y}_{node} - y_i)^2 \\ \hat{y}_{node} = \frac{1}{m_{node}} \sum_{i \in node} y_i \end{cases}$$

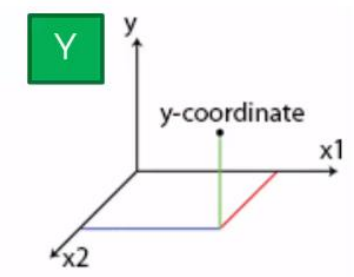
Predicción en nuevos datos

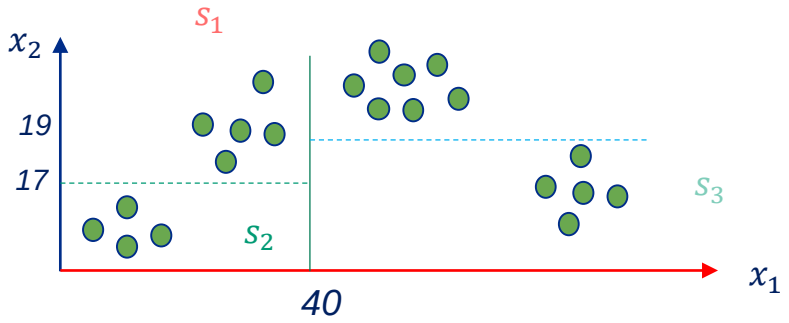
Para predecir el valor de salida para un nuevo punto de datos, sigue el árbol de decisiones basado en las características de ese punto de datos y termina en una de las hojas. El valor predicho es simplemente el promedio de todos los puntos de datos reales en esa hoja.

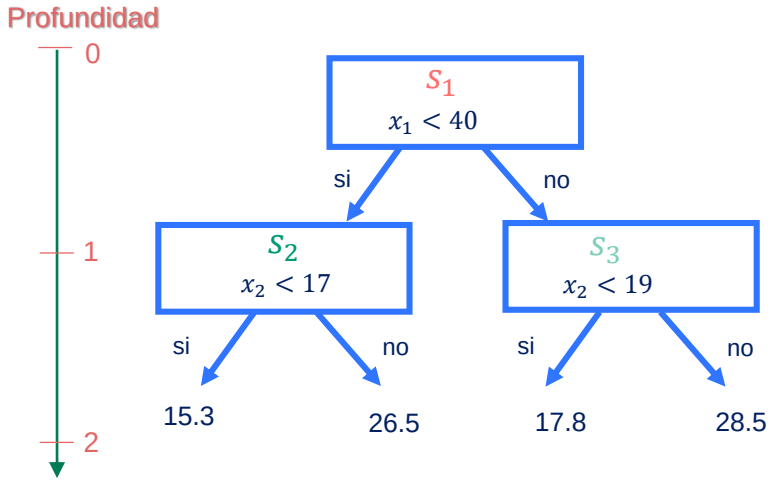
$\hat{y}_{node}$ es el valor objetivo previsto
y_i es el valor del target real
$m_{left/right}$ es el nro. de instancias en el subconjunto $left/right$
k un atributo
t_k un umbral de decisión

Regresión con Decision Trees

Ejemplo



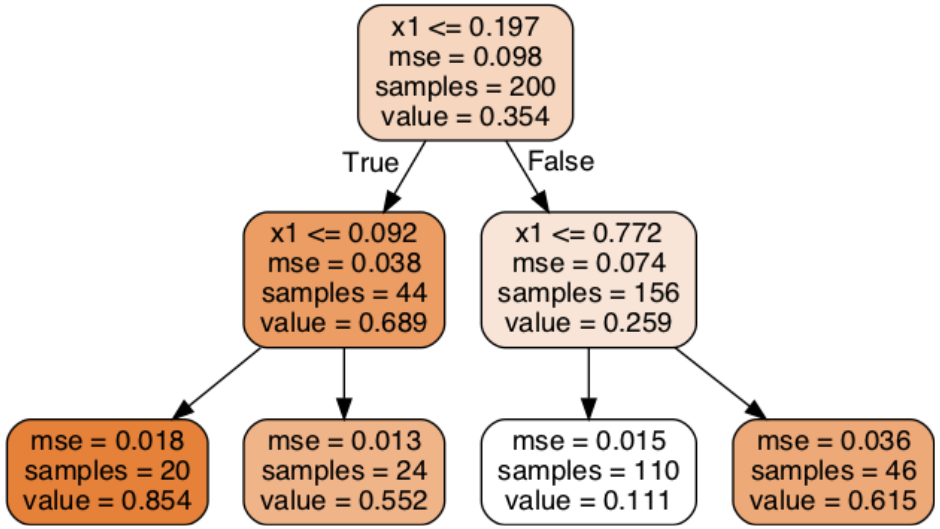




Regresión con Decision Trees

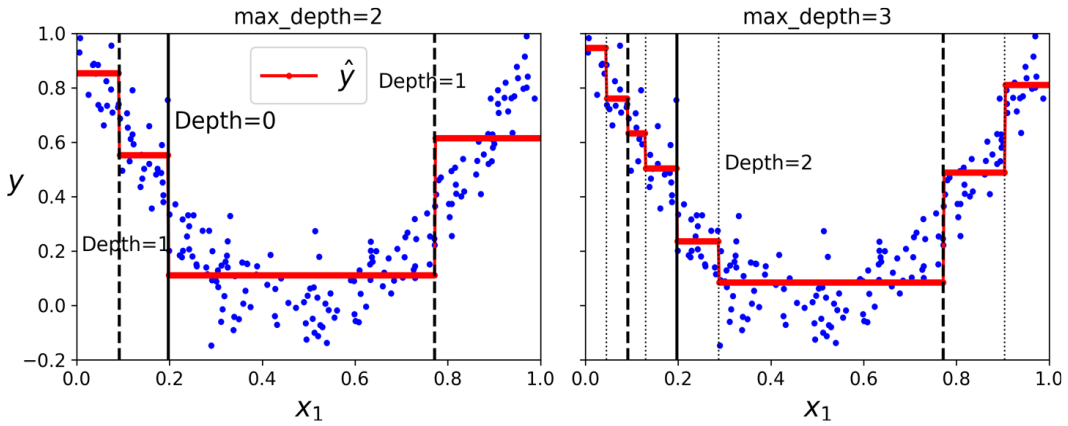
Ejemplo

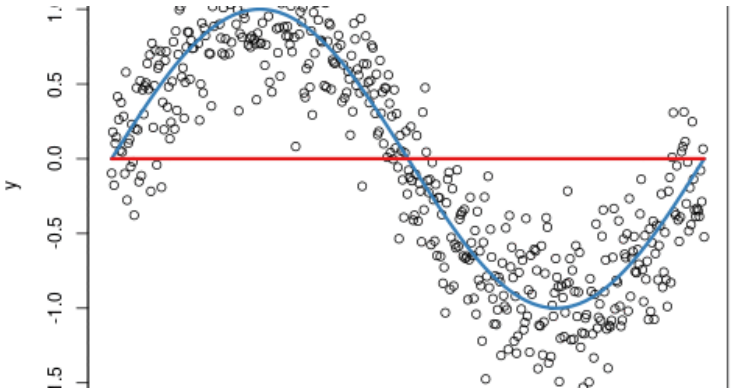
max_depth = 2



Regresión con Decision Trees

Ejemplo: *Predictions of two Decision Tree regression models*





Métricas de Desempeño de Regresión

- $Conj.datos: X_{n\times p}$ $de\ n\ registros;\ p\ atributos;$
- $Un\ regresor\ f(\boldsymbol{x}_i)\ es\ un\ modelo\ que\ predice\ \hat{y}_i\ a\ partir\ de\ \boldsymbol{x}_i$

$y_i =$ $valor\ real;$ $\hat{y}_i$= $target\ predicho$; $\bar{y} = media$

- Principales Métricas del desempeño de un modelo de regresión

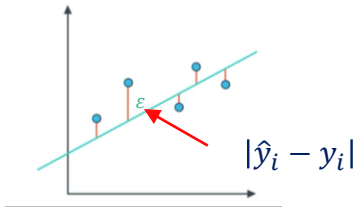
Mean absolute error	$\text{MAE} = \frac{\sum_{i=1}^{n}\lvert y_i - \hat{y}_i\rvert}{n}$
Mean squared error	$\text{MSE} = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$
Coeficiente de determinación	$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$

Coeficiente de determinación (R^2): medida de qué tan bien el modelo acompaña una salida.

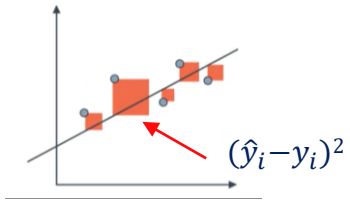
R^2= 0, el modelo está mal ajustado

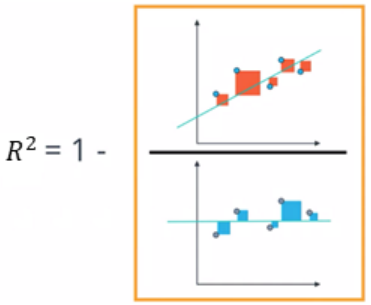
R^2= 1, ajuste perfecto.

- *Mean absolute error*



- *Mean squared error*





sklearn.metrics.r2_score

Bibliografía

- J. Watt and R. Borhani and A. Katsaggelos (2020). Machine Learning Refined: Foundations, Algorithms, and Applications. 2nd Edition. Cambridge: Cambridge University Press.
- C. Bishop (2006). Pattern Recognition and Machine Learning. Springer, New York.
- S. Raschka & V. Mirjalili (2019). *Python Machine Learning*. Third Edition. California: O'Reilly Media.

Sugerencia de links interesantes:

DERIVADAS - Clase Completa: Explicación Desde Cero
https://www.youtube.com/watch?v=_6-zwdrqD3U

DERIVADAS: Las Famosas Reglas EXPLICADAS
https://www.youtube.com/watch?v=O6PeN5SJxzk

¡Gracias!