

# **INTELIGENCIA ARTIFICIAL CON DEEP LEARNING**

**ING. JORGE ALBERTO CASTELLANOS**



Universidad de  
**La Sabana**

---

FACULTAD DE INGENIERÍA

# **¡BIENVENIDOS!**

## **Diplomado en Inteligencia Artificial con Deep Learning**

# Ing. Jorge Alberto Castellanos



- Ingeniero Mecatrónico
- Master en Automatización y robótica
- Profesor por mas de 15 años, en temas de tecnología como robótica y automatización industrial.
- He aplicado los conocimientos de IA en aplicaciones robóticas, en especial las requeridas por la robocup.

# Contenido Parte 1

- **Semana 1**

- Introducción al Aprendizaje de Máquina en Python: entornos de ejecución, librerías, Google Colab.

- **Semana 2**

- Configuración experimental del Aprendizaje de Máquina.
- Procesamiento de Datos
  - Procesamiento de lenguaje natural

- **Semana 3**

- Preprocesamiento de Datos
  - Procesamiento de Lenguaje Natural

- **Semana 4**

- Aprendizaje de Máquina Supervisado
  - Métricas de Desempeño
  - Métodos de Clasificación y Regresión

- **Semana 5 y 6**

- Aprendizaje de Máquina No Supervisado
  - Métricas de Desempeño
  - Métodos de Agrupamiento
  - Reducción de Dimensionalidad

# Cuéntame que sabes o crees de la IA

Conocimientos previos

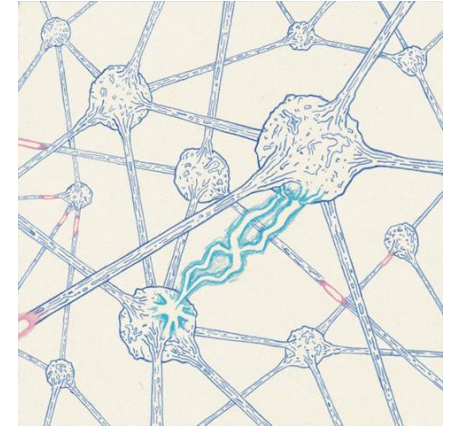


# Qué es la Inteligencia Artificial

- Alan Turing, considerado uno de los padres de la inteligencia artificial, definió la IA como la capacidad de las máquinas para realizar tareas que **requerirían inteligencia humana** si fueran realizadas por un ser humano.

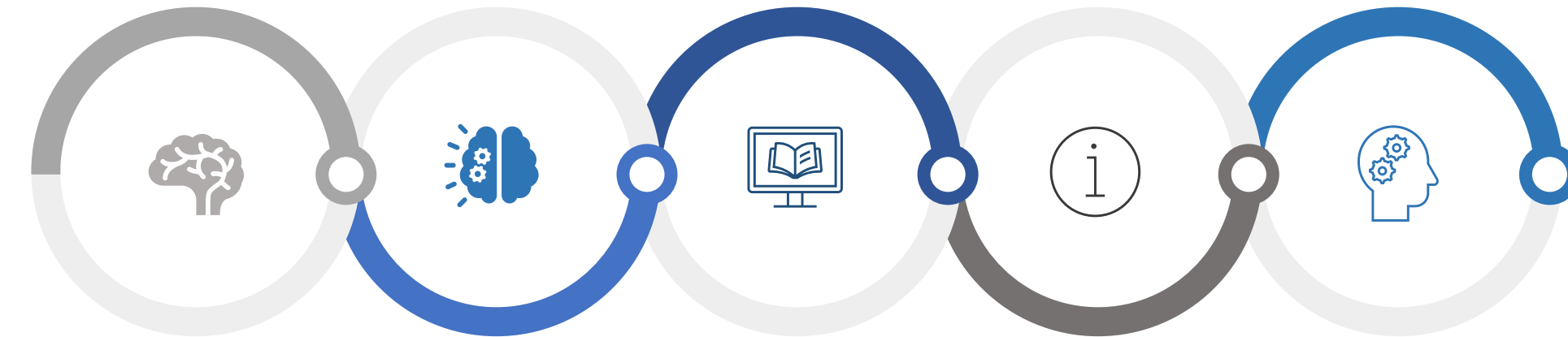


# Qué nos hace inteligentes



Adaptabilidad y  
Versatilidad

Procesamiento  
de Información



Complejidad del  
Cerebro (Aprox  
100M millones)

Uso del  
Simbolismo y el  
Lenguaje

Pensamiento  
Abstracto



# Pequeña historia de la Inteligencia Artificial

## Los Primeros Días

- 1943: McCulloch y Pitts proponen el modelo de circuito booleano del cerebro.
- 1950: Alan Turing publica Computing Machinery and Intelligence e introduce el Test de Turing.

## Enfoques Basados en el Conocimiento

- 1970s: Desarrollo de sistemas basados en reglas (ej., MYCIN, DENDRAL).
- 1980s: Auge de los sistemas expertos en la industria.
- **Finales de 1980s: “Invierno de la IA” debido a expectativas no cumplidas y recortes de financiamiento.**

## Expansión del Aprendizaje Profundo

- 2012: AlexNet revoluciona la visión por computadora, ganando el concurso ImageNet.
- 2016: AlphaGo de DeepMind derrota al campeón mundial de Go, marcando un gran avance en aprendizaje por refuerzo.
- 2018: GPT (Generative Pre-trained Transformer) redefine el procesamiento del lenguaje natural.
- **Finales de 2010s:** Aplicaciones masivas de IA en reconocimiento facial, conducción autónoma y sistemas de recomendación.

1950–1970

1990–2010

1940–1950

1970–1990

2010–2020

## Emoción y Primeros Logros

- 1956: Conferencia de Dartmouth: se acuña el término Inteligencia Artificial.
- 1950s-60s: Primeros programas de IA como el juego de damas de Samuel y el Logic Theorist de Newell y Simon.
- 1965: Robinson desarrolla un algoritmo completo para el razonamiento lógico.

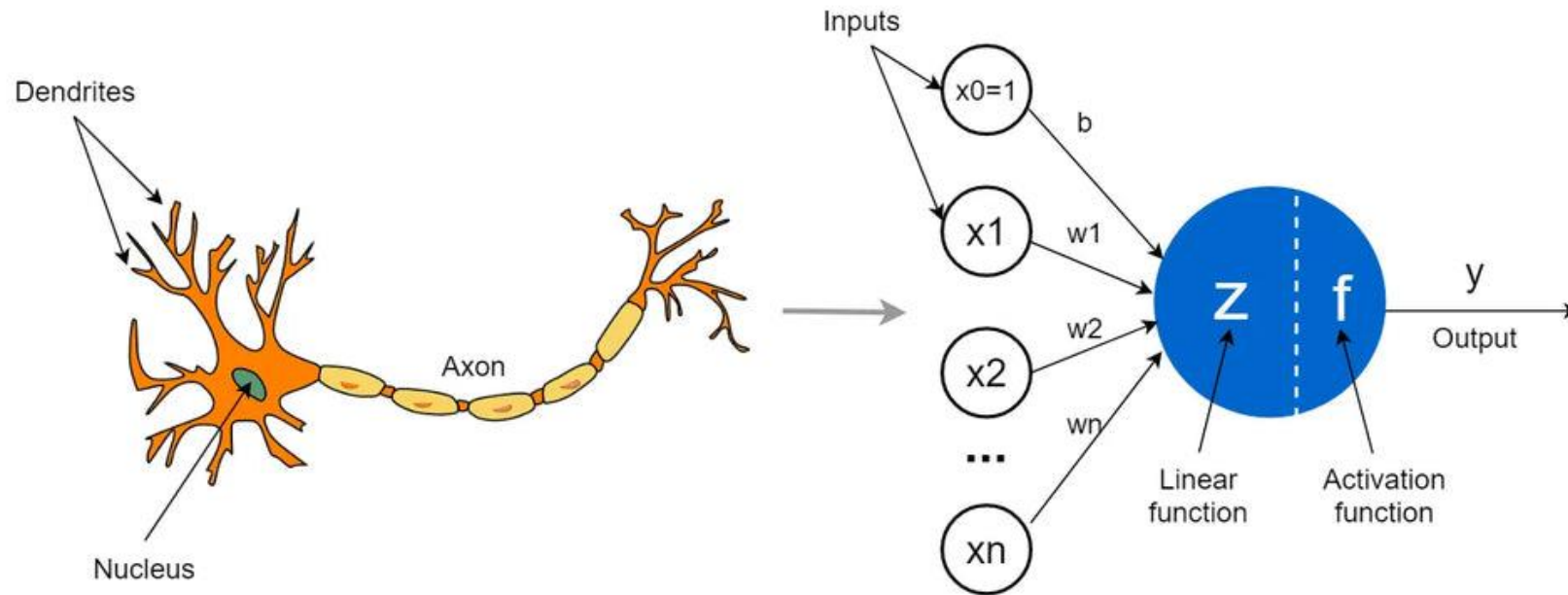
## Enfoques Estadísticos y Renacimiento

- 1990s: Resurgimiento del uso de métodos probabilísticos y algoritmos de aprendizaje.
- 1997: Deep Blue de IBM vence a Garry Kasparov en ajedrez.
- 2006: Geoffrey Hinton populariza el término Deep Learning, impulsando redes neuronales profundas.
- 2010: Auge del aprendizaje supervisado y aplicaciones en visión por computadora, reconocimiento de voz y análisis de datos.

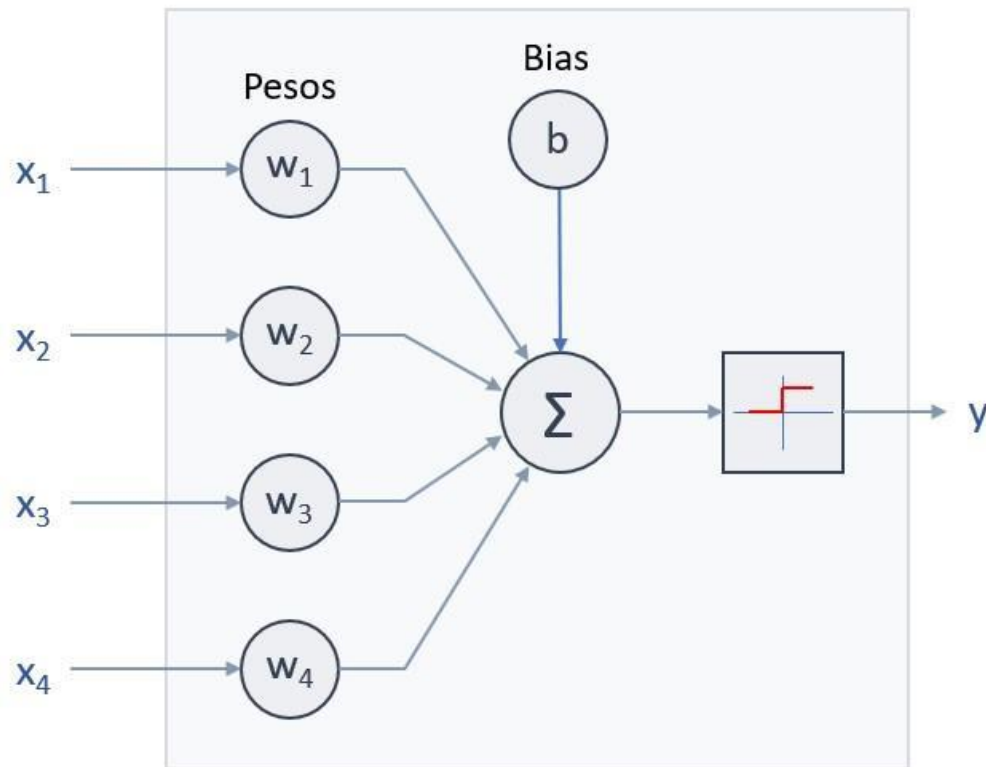


# Unidad básica Perceptrón

Un **perceptrón** es una **neurona artificial** o una unidad básica en el campo de las **redes neuronales**



# Formulación del perceptrón

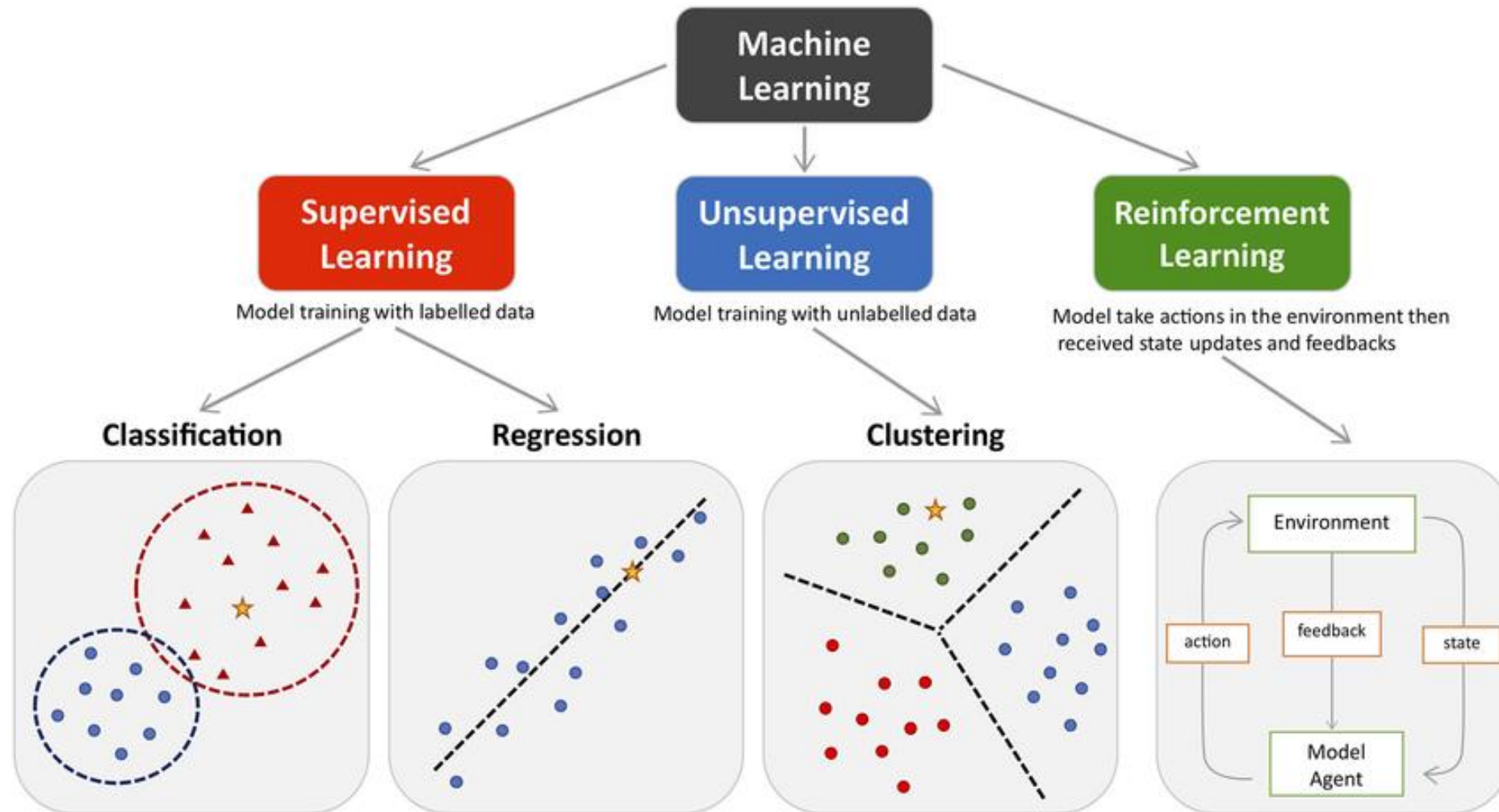


$$salida = \begin{cases} 1 & \text{si } \sum_{i=1}^n w_i \cdot x_i \geq umbral \\ 0 & \text{si } \sum_{i=1}^n w_i \cdot x_i < umbral \end{cases}$$

$$salida = \begin{cases} 1 & \text{si } \sum_{i=1}^n w_i \cdot x_i + b \geq 0 \\ 0 & \text{si } \sum_{i=1}^n w_i \cdot x_i + b < 0 \end{cases}$$

[Video Perceptron](#)

# Introducción al Machine Learning



# Ejemplos Aprendizaje supervisado

1. **Detección de spam en correos electrónicos:** Clasificar correos electrónicos como spam o no spam basándose en características como el contenido del mensaje y el remitente.
2. **Diagnóstico médico:** Predecir enfermedades o condiciones médicas a partir de datos de pacientes, como resultados de pruebas y síntomas.
3. **Reconocimiento de voz:** Convertir el habla en texto, como en asistentes virtuales que entienden y responden a comandos de voz.
4. **Predicción de precios de viviendas:** Estimar el valor de una propiedad basándose en características como la ubicación, el tamaño y el estado.
5. **Detección de fraudes:** Identificar transacciones fraudulentas en tiempo real analizando patrones y comportamientos inusuales en los datos de transacciones.

# Ejemplos Aprendizaje no supervisado

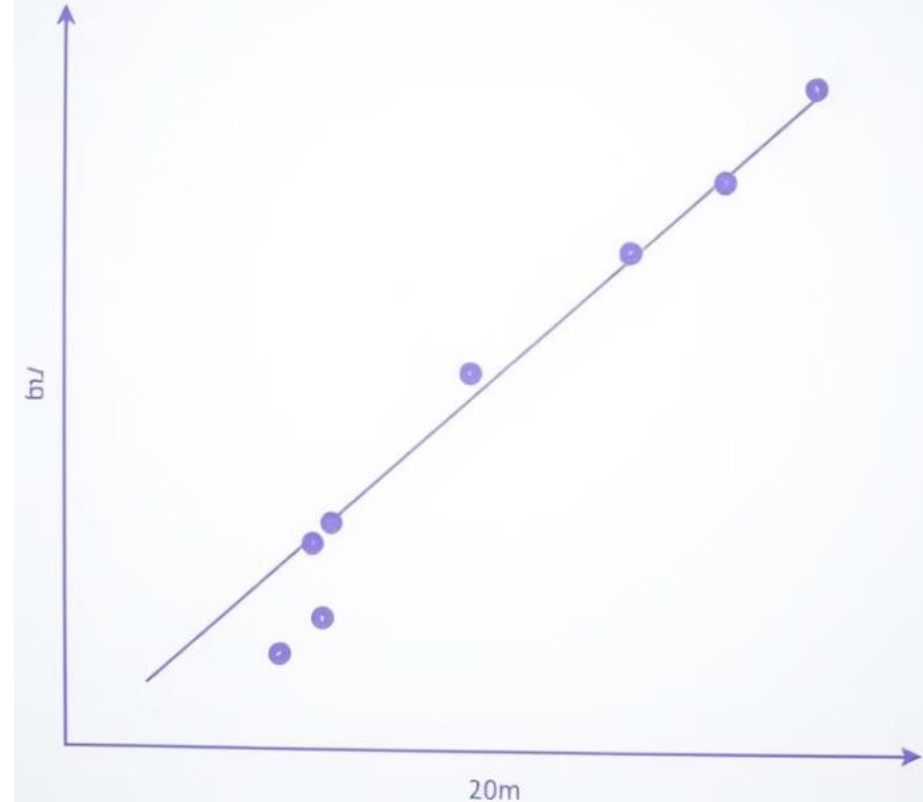
- **Segmentación de clientes:** Agrupar clientes en diferentes segmentos basándose en sus comportamientos de compra y características demográficas.
- **Detección de anomalías:** Identificar patrones inusuales en datos, como fraudes en transacciones financieras o fallos en equipos industriales.
- **Reducción de dimensionalidad:** Simplificar grandes conjuntos de datos manteniendo las características más importantes, como en la compresión de imágenes.
- **Agrupación de documentos:** Organizar documentos en grupos temáticos basándose en el contenido, útil para motores de búsqueda y bibliotecas digitales.
- **Recomendaciones personalizadas:** Ofrecer recomendaciones de productos o contenidos basándose en patrones de comportamiento de usuarios similares.

# Ejemplos Aprendizaje por refuerzo

1. **Juegos:** Algoritmos como AlphaGo y AlphaZero han utilizado aprendizaje por refuerzo para alcanzar niveles de rendimiento sobrehumanos en juegos como Go y ajedrez.
2. **Robótica:** Enseñar a robots a realizar tareas complejas, como caminar, manipular objetos o ensamblar piezas en una línea de producción.
3. **Conducción autónoma:** Desarrollar sistemas de conducción autónoma que puedan tomar decisiones en tiempo real para navegar de manera segura en entornos dinámicos.
4. **Control de semáforos:** Optimizar el flujo de tráfico en intersecciones ajustando los tiempos de los semáforos en función de las condiciones del tráfico en tiempo real.
5. **Gestión de recursos en clústeres informáticos:** Asignar eficientemente recursos en centros de datos para maximizar el rendimiento y minimizar el consumo de energía.

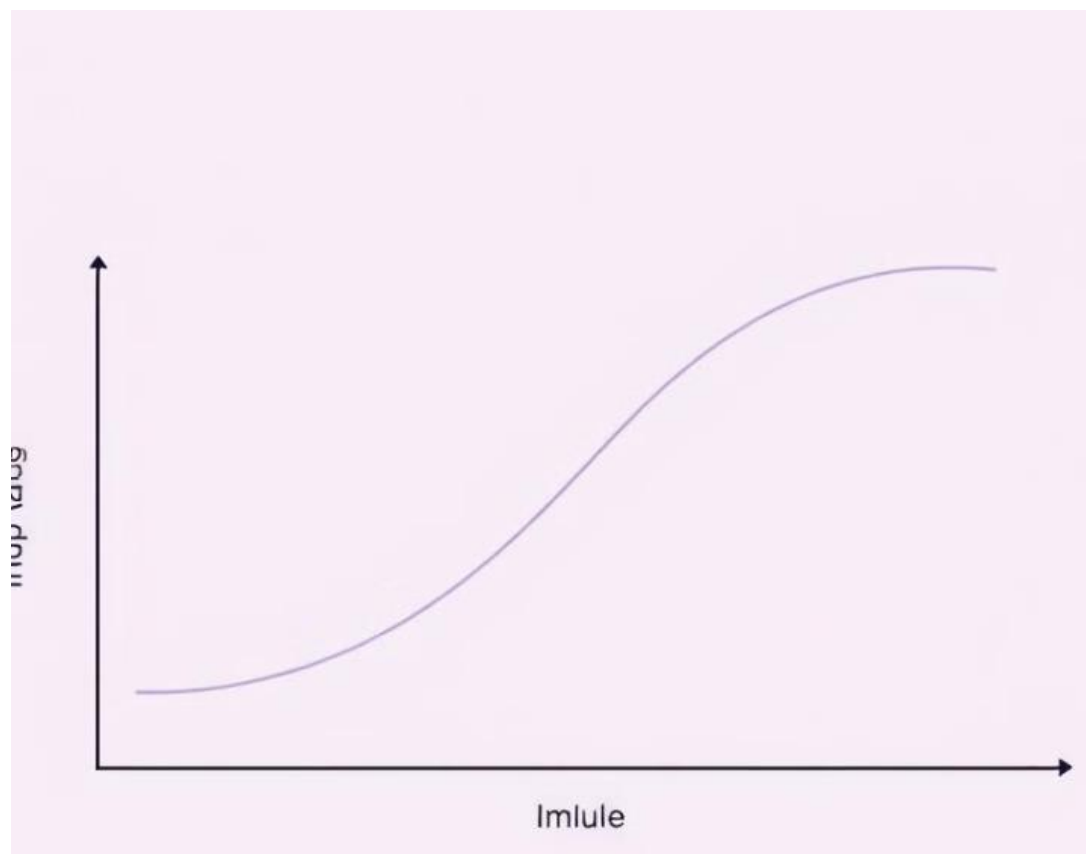
# Regresión Linear

- Predecir una variable objetivo continua basada en una relación lineal con las características de entrada.
- **Gradiente Descendiente:** Optimización de los parámetros del modelo para minimizar la diferencia entre los valores predichos y los valores reales.
- **Regularización:** Prevención del sobreajuste (overfitting) mediante la adición de un término de penalización a la función de pérdida.



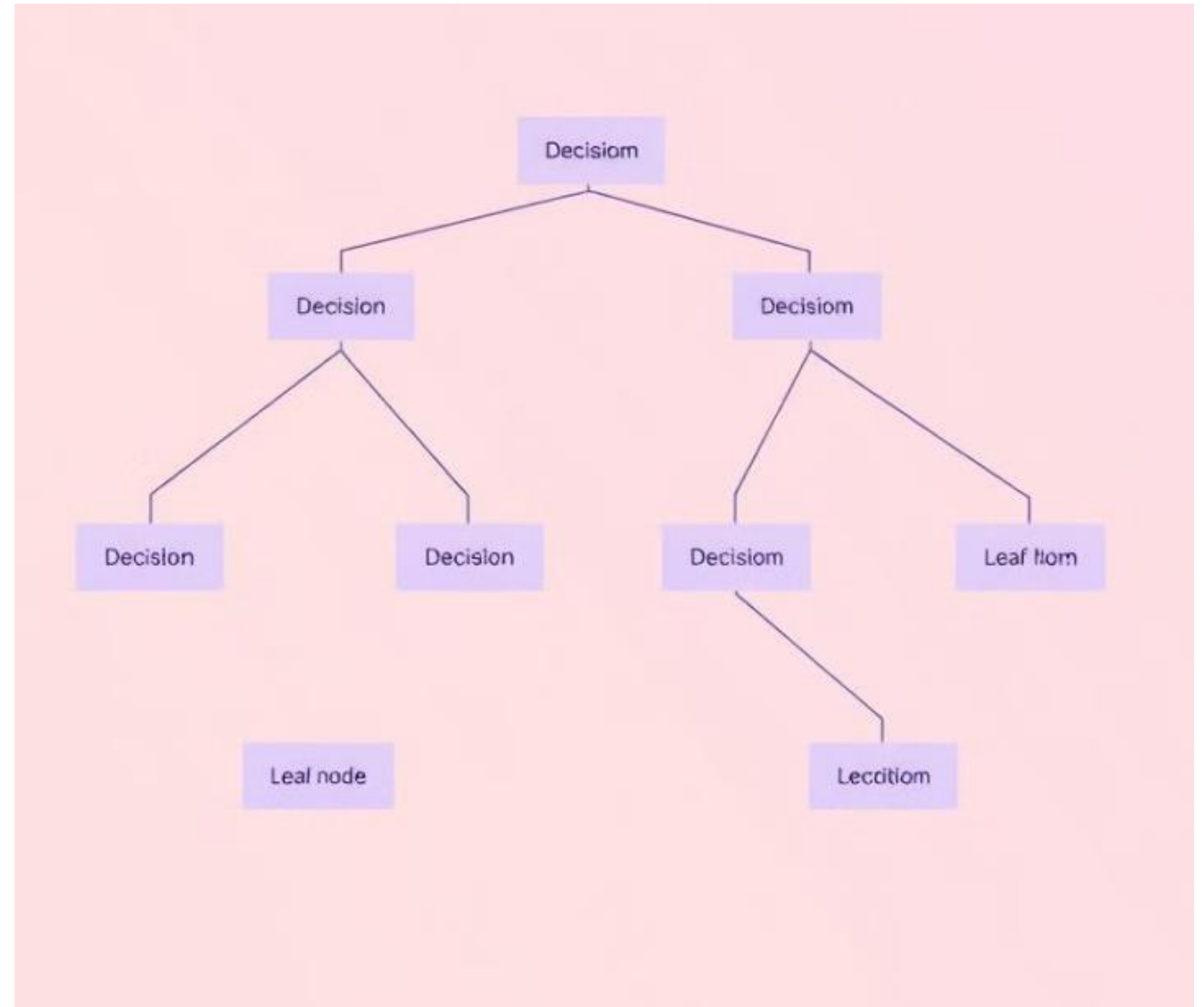


# Regresión Logística



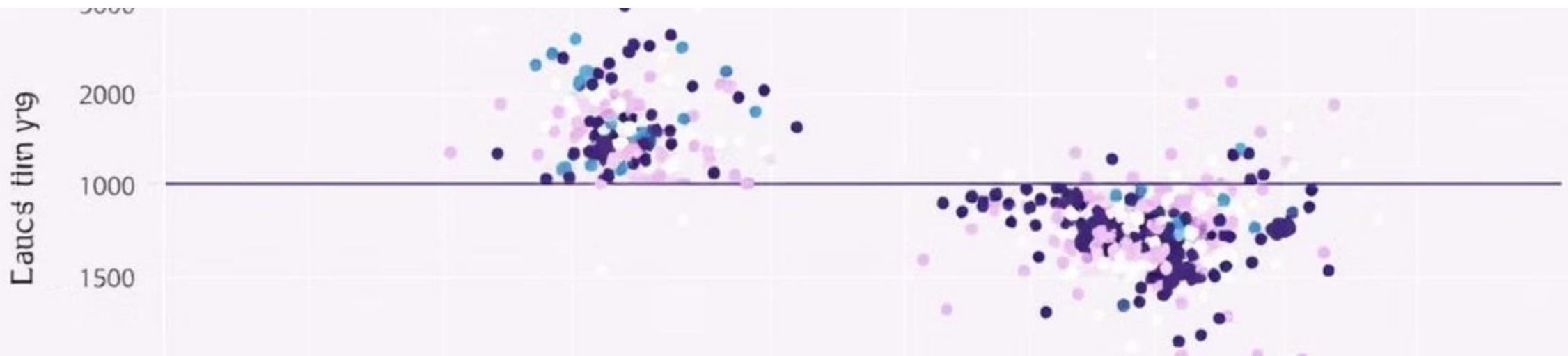
- Predecir la probabilidad de un resultado binario (por ejemplo, sí/no, verdadero/falso).
- **Función de costo:** Medir el rendimiento del modelo y guiar la optimización de parámetros.
- **Aplicaciones:** Detección de spam, diagnóstico médico, predicción de abandono de clientes.

- Modelos basados en árboles que toman decisiones basadas en una serie de reglas.
- **Selección de características:** Elegir las características más informativas para dividir los datos en cada nodo.
- **Aplicaciones:** Segmentación de clientes, evaluación de riesgos, detección de fraudes.



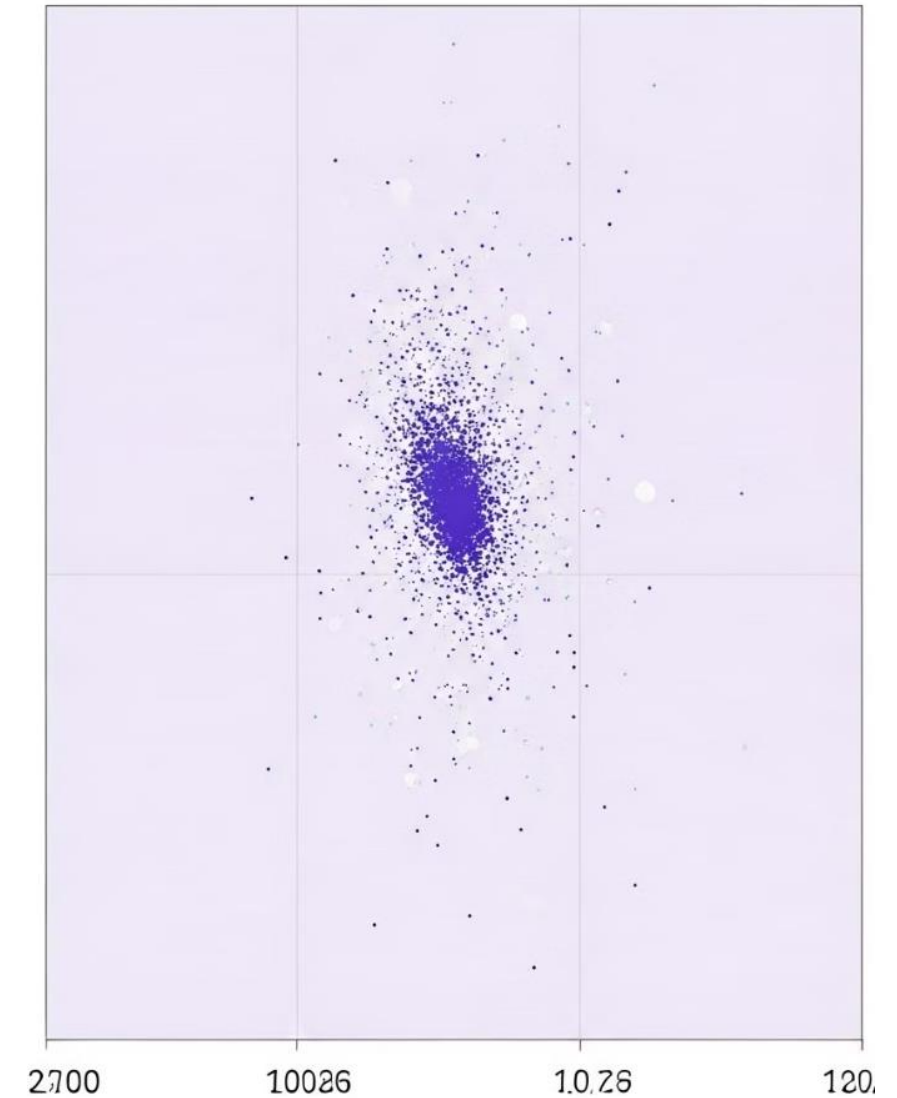
# Máquina de Soporte Vectorial

- Modelos basados en árboles que toman decisiones basadas en una serie de reglas.
- **Selección de características:** Elegir las características más informativas para dividir los datos en cada nodo.
- **Aplicaciones:** Segmentación de clientes, evaluación de riesgos, detección de fraudes.



# K- Vecinos mas cercanos

- Clasificar nuevos puntos de datos basándose en su similitud con vecinos etiquetados.
- **Distancia Métrica:** Medir la similitud entre puntos de datos, como la distancia euclidiana o la distancia de Manhattan.
- **Aplicaciones:** Sistemas de recomendación, reconocimiento de imágenes, detección de anomalías.

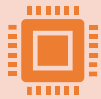


Procesamiento de datos de entrenamiento  
K-NN modelo (s.ite)

# Entornos de ejecución



# ¿Qué son?



**Colab:** Desarrollado por Google, basado en la nube donde se puede programar en Python, con acceso gratuito a GPU y TPU.



**PyTorch:** Desarrollado por Facebook, es muy popular en la comunidad de investigación debido a su flexibilidad y capacidad para realizar cálculos en tiempo real.



**TensorFlow:** Desarrollado por Google, es una plataforma robusta para el desarrollo de modelos de machine learning y deep learning. Ofrece escalabilidad y compatibilidad con múltiples CPU y GPU.



**Jupyter Notebooks:** Aunque no es una librería de machine learning en sí, es un entorno interactivo que permite escribir y ejecutar código en tiempo real, facilitando la experimentación y visualización de datos.



**Keras:** Es una API de alto nivel que se ejecuta sobre TensorFlow y otras plataformas. Es conocida por su simplicidad y facilidad de uso, ideal tanto para principiantes como para expertos.



**Anaconda:** Es una distribución de Python que incluye muchas librerías y herramientas útiles para machine learning y deep learning, como Jupyter Notebooks, pandas, scikit-learn, entre otras.

# Librerías importantes para IA





# Librerías

- **Scikit-learn:** Es una de las librerías más populares para machine learning. Ofrece herramientas simples y eficientes para el análisis de datos y modelado predictivo.
- **TensorFlow:** Desarrollada por Google, es una librería muy utilizada para deep learning. Permite construir y entrenar modelos de redes neuronales de manera flexible y escalable.
- **Keras:** Es una API de alto nivel para redes neuronales, que se ejecuta sobre TensorFlow. Es conocida por su simplicidad y facilidad de uso.
- **PyTorch:** Desarrollada por Facebook, es otra librería muy popular para deep learning. Es especialmente apreciada por su flexibilidad y capacidad para realizar cálculos en tiempo real.
- **Pandas:** Aunque no es específica para machine learning, es esencial para la manipulación y análisis de datos. Facilita la carga, limpieza y preprocesamiento de datos.
- **NumPy:** Fundamental para el cálculo numérico y el manejo de matrices, es ampliamente utilizada en el preprocesamiento de datos y en la implementación de algoritmos de machine learning.
- **Matplotlib:** Utilizada para la visualización de datos, permite crear gráficos y representaciones visuales que ayudan a entender mejor los datos y los resultados de los modelos.

# Colab

**co** Tutorial de Python Diplomado IA.ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at 9:19 AM

+ Code + Text Connecting Gemini

## 1. Conceptos Básicos

Antes de empezar vamos a recorrer algunos conceptos básicos y operaciones que aparecerán en casi todos los códigos que realicen.

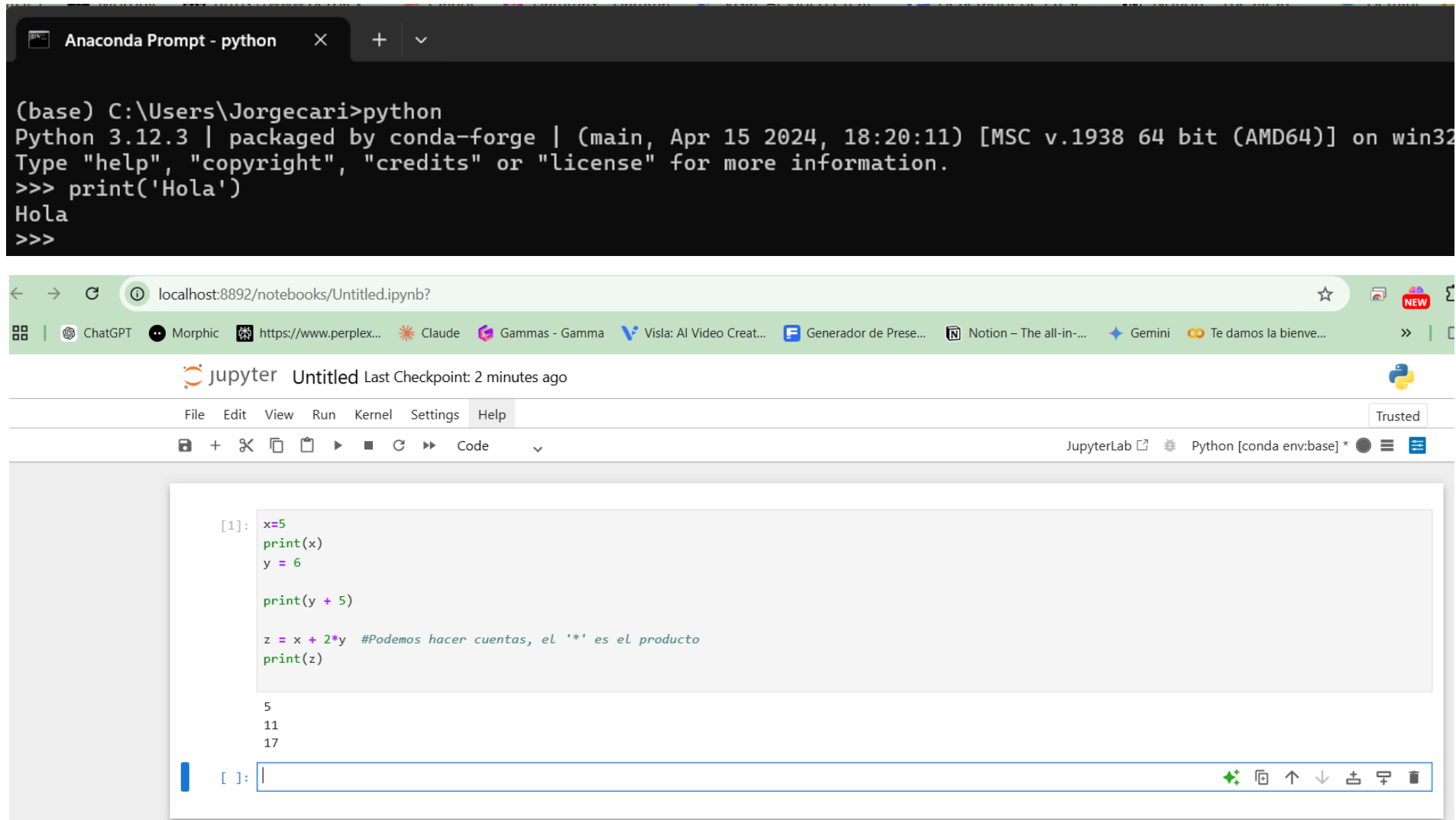
```
x = 5 #Definimos una variable numérica
print(x) #La función 'print()' la escribe en la pantalla

y = 6
print(y + 5) Código a ejecutar

z = x + 2*y #Podemos hacer cuentas, el '*' es el producto
print(z)
```

5  
11  
17 Resultado de ejecución

# Anaconda



The image shows two screenshots related to Anaconda. The top screenshot is an Anaconda Prompt window titled 'Anaconda Prompt - python'. It shows a terminal session where the user has entered 'python' at the prompt, resulting in 'Python 3.12.3 | packaged by conda-forge | (main, Apr 15 2024, 18:20:11) [MSC v.1938 64 bit (AMD64)] on win32'. The user then enters 'Type "help", "copyright", "credits" or "license" for more information.' followed by '>>> print('Hola')', which outputs 'Hola'.

The bottom screenshot is a JupyterLab interface. The browser address bar shows 'localhost:8892/notebooks/Untitled.ipynb?'. The JupyterLab title bar says 'jupyter Untitled Last Checkpoint: 2 minutes ago'. The menu bar includes 'File', 'Edit', 'View', 'Run', 'Kernel', 'Settings', and 'Help'. The toolbar shows various icons for file operations and execution. The main area displays a code cell with the following Python code:

```
[1]: x=5
      print(x)
      y = 6

      print(y + 5)

      z = x + 2*y #Podemos hacer cuentas, el '*' es el producto
      print(z)
```

The output of the code cell is:

```
5
11
17
```

The bottom of the code cell shows a prompt '[ ]: |' and a toolbar with icons for running, saving, and other actions.

# Google Colab

- [Introducción a Google Colab.](#)
- [Introducción a Python con Colab.](#)

