IA local para PyMEs · Módulo 0

Fundamentos y vocabulario

Las palabras con las que se habla el resto del curso, definidas por lo que hacen. Y un glosario con símiles para quien llega de cero.

Qué se construye en este módulo

📖 Glosario con símiles: para entender de qué se habla

Explicar cada término del curso con un símil cotidiano

📖 Texto plano y Markdown: la base de todo

Justificar por qué todo el sistema se construye sobre texto versionable

📖 Modelo, pesos, cuantización, contexto y tokens

Definir cada término de forma medible

Qué se construye en este módulo

📖 **Embeddings, herramienta, agente, skill, harness, MCP y multimodal**

Distinguir cada pieza por lo que hace

0.1 · Glosario con símiles: para entender de qué se habla

Glosario con símiles: para entender de qué se habla

El lunes llega a la ferretería El Tornillo un empleado nuevo. Llamémoslo Modelo. Antes de llegar se aprendió de memoria media biblioteca del mundo: manuales, contratos ajenos, leyes, foros de tornillería. Sabe una barbaridad y contesta rapidísimo.

Doña Marta, la contadora, lo pone a prueba: «¿Cuánto nos debe Tornillería Santander?». Modelo responde con toda seguridad una cifra. Se la inventó.

0.2 · Texto plano y Markdown: la base de todo

Texto plano y Markdown: la base de todo

Todo lo que se construye en este curso —prompts, esquemas, configuraciones, trazas, políticas, documentación de entrega— es **texto**. La decisión más barata y más duradera que se toma al principio es en qué formato vive ese texto. La respuesta es aburrida y correcta: **texto plano**.

Tres razones concretas, no de gusto:

En una tabla

Perfil	Este laboratorio
A · sin GPU	Igual. Es texto
B · 8–16 GB	Igual
C · 24 GB+	Igual

Cuándo NO usar esto

Texto plano y Markdown: la base de todo

- Documentos con diagramación exacta y valor legal (un contrato con márgenes, firmas y foliación): se producen en PDF desde una plantilla; el Markdown es el borra
- Datos tabulares grandes: una tabla Markdown de 2.000 filas no sirve para nada. Eso es CSV o una base de datos; Markdown solo para la tabla resumen.
- Formularios que llena gente no técnica: nadie va a escribir `| celda |` a mano. Se les da una interfaz, y la interfaz guarda texto plano por debajo.

Qué se degrada en cada perfil

A · sin GPU

Igual. Es texto

B · 8–16 GB VRAM

Igual

C · 24 GB+

Igual

0.3 · Modelo, pesos, cuantización, contexto y tokens

Modelo, pesos, cuantización, contexto y tokens

Cada término de esta lección se define por **un número que se puede medir en la terminal**. Si alguien los usa sin número, está hablando de otra cosa.

Un **modelo de lenguaje** es una función enorme que recibe una secuencia de tokens y devuelve, para el siguiente token, una probabilidad por cada token posible del vocabulario. Se muestrea uno, se añade a la secuencia, y se repite. Eso es todo lo que hace. Todo lo demás —«razonar», «resumir», «extraer»— es ese bucle aplicado muchas veces.

En una tabla

Término	El número	Cómo se mide
Modelo (tamaño)	parámetros: 1B, 8B, 70B…	está en el nombre del archivo
Pesos en disco	gigabytes	`ls -lh`
Cuantización	bits por parámetro: 16, 8, 4…	el sufijo: `F16`, `Q8_0`, `Q4_K_M`
Contexto	tokens que caben en la ventana	lo declara el runtime al cargar
Tokens	cuántos pedazos tiene un texto	un programa tokenizador

El comando, copiable

```
cd ~/labs/el-tornillo && mkdir -p labs/00-03 && cd labs/00-03
python3 -m venv .venv && source .venv/bin/activate
pip install -q tokenizers
# tokenizer.json: descárguelo del repositorio del modelo que vaya a usar (Hugging
    Face) y
# póngalo aquí. Es un archivo de texto; no hacen falta los pesos.
```

Si no corre desde cero con un comando, no entra al curso.

Cuándo NO usar esto

Modelo, pesos, cuantización, contexto y tokens

- No cuente tokens con el tokenizador de otro modelo para presupuestar contexto: las cifras pueden diferir un 30 %.
- No cuantice a Q2/Q3 para «que quepa» en un equipo que no da: mida primero; casi siempre es mejor un modelo más pequeño en Q8 que uno grande en Q2.
- No compre por parámetros. Un 70B en Q2 puede rendir peor que un 8B en Q8 y costar cuatro veces más de memoria.

Qué se degrada en cada perfil

A · sin GPU

Igual: tokenizar es CPU

B · 8–16 GB VRAM

Igual

C · 24 GB+

Igual

0.4 · Embeddings, herramienta, agente, skill, harness, MCP y multimodal

Embeddings, herramienta, agente, skill, harness, MCP y multimodal

Esta lección cierra el vocabulario con las piezas que **actúan**. Cada una se define por lo que hace y se demuestra con el ejemplo más pequeño que corre. Nada de esto necesita GPU.

Un **embedding** es un vector de números que representa el significado de un texto, de modo que textos parecidos tienen vectores cercanos. Lo produce un **modelo de embeddings**, distinto del modelo de lenguaje y mucho más pequeño.

El comando, copiable

```
# Un servidor MCP mínimo que expone buscar_factura. [VOLÁTIL — verificado: 2026-09]
pip install -q mcp
```

Si no corre desde cero con un comando, no entra al curso.

Cuándo NO usar esto

Embeddings, herramienta, agente, skill, harness, MCP y multimodal

- Un agente para una tarea que un flujo determinista resuelve (mover un archivo, renombrar, sumar una columna). Módulo 9: el modelo solo donde hay ambigüedad.
- Embeddings para buscar un número exacto (una cédula, un número de factura). Eso es una búsqueda exacta en base de datos; los embeddings traerán «parecidos».
- MCP para una sola herramienta que usa un solo programa. Es un enchufe: vale la pena cuando hay más de un aparato.

Entregables del módulo

- el `README.md` versionado. De aquí en adelante, cada laboratorio termina con un commit cuyo mensaje empieza por «Capa N».
- `labs/00-03/tokens.py` y una tabla con los tokens de los tres documentos más largos del cliente.

Las palabras con las que se habla el resto del curso, definidas por lo que hacen.