

# Manual de diseño de CanBot

Chatbot híbrido para la recomendación de experiencias turísticas en Cantabria

## 1. Introducción

### 1.1 Objetivo del manual

Describir el diseño, la arquitectura, el rendimiento y el funcionamiento de **CanBot**, chatbot para recomendar experiencias turísticas en Cantabria.

### 1.2 Descripción general

**CanBot** es un asistente conversacional híbrido de alta modularidad que ofrece dos variantes, según las necesidades y requisitos específicos del cliente.

**a) Modelo CanBot Veloz:** sistema híbrido que combina técnicas clásicas de clasificación, basadas en Machine Learning (**regresión logística**) con un LLM (Large Language Model).

**b) Modelo CanBot Infalible:** sistema híbrido, basado en Aprendizaje Profundo, con una red neuronal entrenada con embeddings contextuales combinado con un LLM.

De este modo, **CanBot** le ofrece a su empresa la posibilidad de elegir entre la combinación de un clasificador rápido, eficiente y clásico con un modelo generativo del lenguaje (LLM) de última generación y otro, con una excelente comprensión semántica y alto rendimiento, combinado con el mismo LLM.

Ambos modelos guiarán al usuario, paso a paso, para elegir experiencias turísticas personalizadas en Cantabria, según sus intereses: naturaleza, cultura, gastronomía, espectáculos en vivo y compras.

## 2. Características generales del sistema CanBot

Funcionalidad	Estado
Clasificación multietiqueta con alto rendimiento	ML, Regresión Logística
Clasificación multietiqueta con máxima precisión	Red Neuronal + embeddings Sentence-BERT en español
Generación de recomendaciones turísticas	Generado por LLM
Evaluación híbrida automática	Funcional
Adaptación a nuevos dominios	Escalable
Bajo coste computacional inicial	Optimizado
Posibilidad de integración o sustitución entre sistemas	Aportado
Modularidad y escalabilidad	Garantizada
Documentación	Incluida

## 3. Público objetivo

### 3.1 Quién usará CanBot

El público objetivo son turistas o residentes que busquen recomendaciones turísticas en Cantabria.

## 4. Arquitectura general

### 4. 1. Árbol de decisión

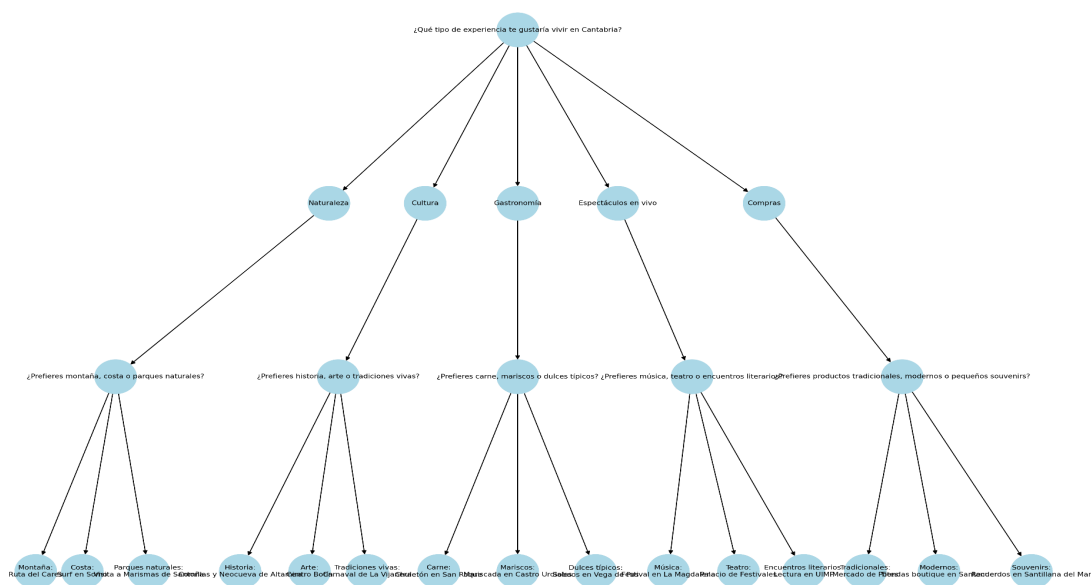
Un árbol de decisión es un algoritmo de aprendizaje supervisado empleado para la clasificación.

Tiene una estructura jerárquica que consta de un nodo raíz, ramas, nodos internos y hojas.

Su estructura es adecuada para la creación de un consultor de viajes breves, como **CanBot**, ya que permite guiar al usuario en la elección de una experiencia turística personalizada en Cantabria, según sus preferencias, mediante la invocación de una función recursiva.

A partir de la primera elección, el árbol presenta 5 nodos que, a su vez, se subdividen en 15 subnodos por categoría que desembocan en una recomendación final.

La estructura simétrica del árbol de decisión en que **CanBot** se basa es adecuada para generar un corpus equilibrado y, de ese modo, evitar sesgos en el entrenamiento de los modelos de clasificación.



**Figura 1.** Esquema estructural del árbol de decisión para la recomendación de experiencias turísticas en Cantabria.

A continuación, se incluye la **definición del árbol de decisión** en formato de diccionario Python:

```
decision_tree = {  
  
    "question": "¿Qué tipo de experiencia te gustaría vivir en Cantabria?",  
  
    "options": {  
  
        "naturaleza": {  
  
            "question": "¿Prefieres montaña, costa o parques naturales?",  
  
            "options": {  
  
                "montaña": "recomienda: Ruta del Cares en el límite con Asturias",  
  
                "costa": "recomienda: Surf en Somo con tabla alquilada",  
  
                "parques naturales": "recomienda: Visita ornitológica a las Marismas de Santoña."  
  
            }  
  
        },  
  
        "cultura": {  
  
            "question": "¿Prefieres historia, arte o tradiciones vivas?",  
  
            "options": {  
  
                "historia": "recomienda: Ruta guiada a Comillas y Neocueva de Altamira",  
  
                "arte": "recomienda: Centro Botín en Santander",  
  
                "tradiciones vivas": "recomienda: Visita el carnaval de La Vijanera"  
  
            }  
  
        },  
  
        "gastronomía": {  
  
            "question": "¿Prefieres carne, mariscos o dulces típicos?",
```

```

"options": {

    "carne": "recomienda: Chuletón en un restaurante de San Roque",

    "mariscos": "recomienda: Mariscada en Castro Urdiales",

    "dulces típicos": "recomienda: Sobaos y quesadas en Vega de Pas"

}

},

"espectáculos en vivo": {

    "question": "¿Prefieres música, teatro o encuentros literarios?",

    "options": {

        "música": "recomienda: Festival de verano en la campa de La Magdalena",

        "teatro": "recomienda: Obra clásica en el Palacio de Festivales",

        "encuentros literarios": "recomienda: Lectura o presentación de autor en la
Universidad Internacional Menéndez Pelayo."

    }

},

"compras": {

    "question": "¿Prefieres productos tradicionales, modernos o pequeños souvenirs?",

    "options": {

        "tradicionales": "recomienda: Miel, quesos y sobaos en mercado de Potes",

        "modernos": "recomienda: Diseño local en tiendas boutique de Santander",

        "souvenirs": "recomienda: Pequeños recuerdos típicos en las tiendas de Santillana
del Mar."

    }

}

```

}

}

}

## **4. 2. Ejemplos de diálogos por nodo de las cinco categorías principales**

### **a) Nodo: naturaleza**

#### **Diálogo 1:**

-Usuario: Me gustaría hacer senderismo o disfrutar del paisaje natural.

-Chatbot: ¡Perfecto! Te recomiendo una experiencia en la naturaleza como la Ruta del Cares en el límite con Asturias.

### **b) Nodo: cultura**

#### **Diálogo 2:**

-Usuario: Busco un plan relacionado con el arte o la historia.

-Chatbot: Entonces podrías disfrutar de un recorrido cultural por Santander o visitar exposiciones locales.

### **c) Nodo: gastronomía**

#### **Diálogo 3:**

-Usuario: Me interesa probar platos de cocina tradicionales.

-Chatbot: ¡Buena idea! Puedes disfrutar de un buen chuletón en San Roque.

### **d) Nodo: espectáculos**

#### **Diálogo 4:**

-Usuario: Me gustaría escuchar poesía en un ambiente tranquilo.

-Chatbot: Estaría bien acudir a un encuentro literario en la Universidad Internacional Menéndez Pelayo.

**e) Nodo: compras**

**Diálogo 5:**

-Usuario: Me interesan los productos artesanales locales elaborados con buenas materias primas.

-Chatbot: Te sugiero visitar el mercado de Potes con productos como miel, quesos o sobaos.

### **4. 3. Flujo de decisión híbrido. Elige la opción que mejor se adapta a tu necesidad: CanBot Veloz/CanBot Infalible**

El flujo de **CanBot** se estructura de la siguiente manera:

**1. Clasificación inicial (nivel 1)**

a) **Con TF-IDF (CanBot Veloz):** la entrada del usuario se vectoriza con el modelo TF-IDF y se clasifica mediante un modelo de regresión logística (clf\_root).

b) **Con Sentence-BERT (CanBot Infalible):** la entrada del usuario se convierte en embeddings semánticos con el modelo especializado de Sentence-BERT, hiiamsid/sentence\_similarity\_spanish\_es y se clasifica mediante una red neuronal sencilla entrenada sobre estos embeddings.

**2. Evaluación de confianza:** si la predicción supera un umbral de confianza (0.6) y pertenece a una de las categorías definidas, se considera válida.

**3. Generación con LLM (nivel 2):** el modelo LLM (Meta-Llama-3.1-8B-Instruct) se activa automáticamente en el segundo nivel. Además, si la confianza es insuficiente o la entrada es difícil de clasificar, actúa directamente como intérprete principal de la intención del usuario (*fallback total*).

## **5. Dataset, entrenamiento y evaluación de los modelos del primer nivel jerárquico**

### **5. 1. Dataset**

Para entrenar los clasificadores se ha construido un corpus etiquetado generado sintéticamente, compuesto por ejemplos representativos de intenciones turísticas en Cantabria.

El corpus está diseñado de un modo balanceado que garantiza una distribución equitativa entre clases y minimiza el sesgo en el entrenamiento del modelo.

#### **5. 1. 1. Estructura general del corpus**

##### **Categorías principales (nivel 1):**

- Naturaleza
- Cultura
- Gastronomía
- Espectáculos en vivo
- Compras

##### **Subcategorías (nivel 2):**

- Naturaleza: Montaña, Costa, Parques Naturales
- Cultura: Historia, Arte, Tradiciones vivas
- Gastronomía: Carne, Mariscos, Dulces típicos
- Espectáculos en vivo: Música, Teatro, Encuentros literarios
- Compras: Tradicionales, Modernos, Souvenirs

#### **5. 1. 2. Resumen global del corpus**

- Categorías principales: 5
- Frases por categoría principal: 70
- Subcategorías: 15
- Frases por subcategoría: 30 frases
- Total de frases en el corpus: 800



### 5. 1. 3. Etiquetas y configuración

Cada frase está etiquetada con un nombre identificativo de su categoría o subcategoría para entrenar al modelo supervisado.

Algunas frases etiquetadas en las **categorías principales del nivel 1** (como “Naturaleza”, “Cultura” o “Gastronomía”) incluyen intencionadamente contenido semántico propio de las **subcategorías de nivel 2** (por ejemplo, “Montaña”, “Historia” o “Marisco”) para facilitar que el modelo de regresión logística o, en su caso, la red neuronal entrenada con embeddings contextuales generados por el modelo afinado de Sentence-BERT, identifiquen intenciones generales a partir de expresiones más específicas.

## 5. 2. Preprocesamiento de entradas y entrenamiento

### 5. 2. 1. Modelo de Regresión Logística

#### a) Preprocesamiento

El corpus se ha transformado en vectores numéricos mediante la técnica **TF-IDF (Term Frequency – Inverse Document Frequency)**.

El vectorizador empleado, **TfidfVectorizer**, perteneciente al módulo `sklearn.feature_extraction.text`. se ha configurado con los siguientes parámetros de normalización textual:

- `lowercase=True`
- `strip_accents=unicode`

Este proceso ha producido una **matriz dispersa de dimensiones (800 x 1099)**, que constituye una entrada para el modelo de clasificación supervisado con una buena cobertura léxica y semántica.

#### b) Entrenamiento del clasificador del primer nivel (ML)

Tras la vectorización del corpus, se ha entrenado un modelo de regresión logística para clasificar las frases del usuario en una de las cinco categorías generales del árbol de decisión.

Mediante una partición estratificada, con un 80 % de los datos destinados al entrenamiento y un 20 % a la evaluación, en la que cada clase se ha representado equitativamente, se ha entrenado un modelo con **LogisticRegression** de la biblioteca **scikit-learn**, configurada

con un máximo de 1000 iteraciones para asegurar la convergencia del algoritmo en todos los casos.

El modelo ha sido posteriormente validado en el conjunto de prueba para verificar su capacidad de generalización.

### **5.2.2. Modelo con arquitectura de Red Neuronal con embeddings basados en Sentence-BERT**

#### **a) Preprocesamiento**

En este caso, el corpus ha sido transformado en representaciones numéricas mediante embeddings contextuales generados por un modelo afinado de Sentence-BERT en español, `hiiamsid/sentence_similarity_spanish_es`, basado en una arquitectura Transformer encoder.

Este proceso proporciona representaciones vectoriales densas que capturan información semántica profunda en una matriz numérica con dimensiones (800 x 768), que ofrece una base discriminativa para la clasificación.

#### **b) Entrenamiento del clasificador del primer nivel (Red Neuronal con embeddings basados en Sentence-BERT)**

Una vez obtenidas las representaciones vectoriales del corpus, se ha procedido a entrenar una red neuronal sencilla, estructurada con dos capas ocultas (20 neuronas en la primera capa y 10 en la segunda, ambas con la función de activación ReLU).

Para la clasificación multiclase de las cinco categorías generales, se ha seleccionado una capa de salida con función de activación softmax.

Como en el caso anterior, se ha realizado una partición estratificada del corpus (80% para entrenamiento y un 20% para evaluación). El entrenamiento se ha desarrollado durante 50 épocas, con el optimizador Adam y una función de pérdida categórica (cross-entropy).

La validación continua durante el entrenamiento ha asegurado que el modelo optimice adecuadamente la capacidad de generalización y minimice el sobreajuste.

### 5. 3. Evaluación

La evaluación de los modelos entrenados del primer nivel jerárquico se ha realizado sobre el conjunto de prueba, compuesto por el 20 % del total de frases del nodo raíz.

Se han calculado las métricas estándar de clasificación para cada una de las cinco categorías principales. Las métricas utilizadas han sido:

**Precisión:** proporción de predicciones correctas entre todas las realizadas para una clase determinada.

**Exhaustividad (recall):** proporción de casos correctamente identificados entre todos los ejemplos reales de esa clase.

**Puntuación F1 (F1-score):** media armónica entre precisión y recall, que resume la calidad general del modelo por clase.

**Tabla 1. Evaluación del clasificador del nodo raíz (modelo LogisticRegression)**

#### **Métricas generales (macro)**

<b>Métrica</b>	<b>Valor</b>
<b>Accuracy</b>	<b>0.814</b>
<b>Precision macro</b>	<b>0.838</b>
<b>Recall macro</b>	<b>0.814</b>
<b>F1-score macro</b>	<b>0.817</b>

## Reporte por Clase

Clase	Precisión	Recall	F1-score	Soporte
Compras	1.000	0.929	0.963	14
Cultura	0.706	0.857	0.774	14
Espectáculos en vivo	0.909	0.714	0.800	14
Gastronomía	0.667	0.857	0.750	14
Naturaleza	0.909	0.714	0.800	14
Exactitud total (Accuracy)			0.814	70
Macro promedio	0.838	0.814	0.817	70
Promedio ponderado	0.838	0.814	0.817	70

Tabla 2. Evaluación del clasificador del nodo raíz (modelo Red Neuronal con embeddings, hiiasid/sentence\_similarity\_spanish\_es)

## Métricas generales (macro)

Métrica	Valor
Accuracy	0.9429
Precision macro	0.9448
Recall macro	0.9429
F1-score macro	0.9433

## Reporte por Clase

Clase	Precisión	Recall	F1-score	Soporte
Compras	0.929	0.929	0.929	14
Cultura	1.000	0.929	0.963	14
Espectáculos en vivo	0.867	0.929	0.897	14
Gastronomía	0.929	0.929	0.929	14
Naturaleza	1.000	1.000	1.000	14
Exactitud total (Accuracy)			0.943	70
Macro promedio	0.945	0.943	0.943	70
Promedio ponderado	0.945	0.943	0.943	70

## 6. Técnicas descartadas

Durante el proceso de creación de **Canbot** se han entrenado y evaluado otros modelos que ha sido descartados por su bajo rendimiento:

- Un modelo de red neuronal simple (20-10 capas, con activación ReLU) entrenado sobre representaciones TF-IDF.
- El modelo basado en embeddings contextuales generados por SBERT (hiiamsid/sentence\_similarity\_spanish\_es) entrenado para clasificar 20 categorías.

En el primer caso, la red neuronal ha mostrado un rendimiento deficiente, con una precisión media y un macro-F1 inferiores al 70 %, y una marcada incapacidad para diferenciar clases semánticamente próximas, debido al tamaño insuficiente del dataset y a la falta de contexto inherente a la representación TF-IDF.

Por su parte, el modelo basado en SBERT, si bien ha alcanzado métricas excelentes en la clasificación de las cinco categorías principales, en la categorización de las 15 subcategorías y 5 clases raíz (debido probablemente al solapamiento léxico multiclase) ha experimentado un menor rendimiento y su F1-score macro ha descendido a 0.635, mientras que su precisión global apenas ha alcanzado el 63.7 %.

En consecuencia, ambos modelos han sido considerados insuficientes para garantizar una clasificación fiable y se han descartado de su implementación en **CanBot**.

## **7. Dataset, entrenamiento y evaluación del modelo generador del segundo nivel (LLM)**

El segundo nivel del sistema CanBot se ha desarrollado mediante un modelo generativo basado en LLM (Large Language Model), afinado específicamente para generar recomendaciones turísticas a partir de ejemplos conversacionales reales.

Este modelo complementa el árbol estructurado del primer nivel en situaciones en las que se requiere una mayor comprensión semántica y una respuesta personalizada.

### **7. 1. Corpus de afinado (fine-tuning)**

El modelo se ha ajustado mediante *fine-tuning* a través de un corpus compuesto por diálogos sintéticos, generados manualmente a partir de las subcategorías del árbol de decisión: naturaleza (montaña, costa, parques), cultura (historia, arte, tradiciones), gastronomía (carne, mariscos, dulces), espectáculos en vivo (música, teatro, literatura) y compras (tradicionales, modernos, souvenirs).

Cada diálogo consta de una intención expresada por el usuario y una recomendación formulada por el asistente, siguiendo un patrón conversacional realista.

El conjunto completo incluye 150 pares de diálogos, distribuidos equitativamente entre las subcategorías.

### **7. 2. Modelo utilizado**

Se ha empleado el modelo Meta-Llama-3.1-8B-Instruct, cargado en formato 4-bit para optimizar el rendimiento y consumo de memoria en GPU.

Los principales datos del modelo son:

- **Nombre del modelo:** unsloth/Meta-Llama-3.1-8B-Instruct-bnb-4bit

- **Longitud máxima de secuencia:** 1024 tokens.
- Cargado con **optimización de cuantización** 4-bit para reducir recursos computacionales
- **Plantilla de conversación:** llama-3.1, aplicada para estructurar los turnos de diálogo user ↔ assistant.

### 7. 3. Proceso de entrenamiento

El proceso de ajuste fino se ha llevado a cabo con la biblioteca trl (Transformers Reinforcement Learning), a través de los siguientes pasos:

1. Conversión del corpus a formato Dataset compatible con Hugging Face.
2. Aplicación de la plantilla llama-3.1 con `apply_chat_template`.
3. Conversión a texto plano usando una función `to text()` para formar las entradas de entrenamiento.
4. Entrenamiento con SFTTrainer (Supervised Fine-Tuning), mediante:
  - Tasa de aprendizaje: 2e-4
  - Batch size: 4
  - Máximo de pasos: 260
  - Entrenamiento en precisión mixta (fp16)
  - Output: carpeta outputs/ con checkpoints.

### 7. 4. Evaluación cualitativa del modelo

Además del entrenamiento supervisado, se ha realizado una evaluación cualitativa del LLM mediante pruebas manuales con frases deliberadamente ambiguas, subjetivas o fuera del estilo clásico del corpus.

El objetivo ha sido verificar la capacidad del modelo para mantener la coherencia temática, generar recomendaciones precisas y adaptarse a solicitudes abiertas.

Los resultados muestran que el modelo **unsloth/Meta-Llama-3.1-8B-Instruct-bnb-4bit** responde de forma pertinente, incluso cuando el usuario no menciona explícitamente una categoría o subcategoría.

A continuación, se muestran algunos ejemplo representativos de diálogos usuario-**CanBot**:

Entrada del usuario	Respuesta generada (LLM, CanBot)
"Algo tranquilo para pasear cerca del agua"	"Pasea junto al río Pas, ideal para desconectar entre naturaleza e historia".
"Quiero disfrutar de algo relajante y salvaje a la vez"	"Visita el Parque de Urdón y disfruta del aire libre entre los hayedos".
"Me encanta lo abstracto"	"En Soba, el Museo de la Madera y del Papel es un espacio minimalista pero impactante."

Estas pruebas demuestran que el modelo LLM, tras el ajuste fino, es capaz de generar respuestas coherentes y adecuadas al tono del usuario, incluso cuando la intención no está bien definida.

## 7. 4 Integración del modelo LLM en el flujo del chatbot

El modelo generativo de segundo nivel (LLM) se ha integrado de forma complementaria al clasificador del primer nivel (bien junto al modelo de regresión logística, bien combinado con la red neuronal con embeddings).

Su función principal es generar recomendaciones personalizadas cuando la categoría general ya ha sido detectada con suficiente confianza, la entrada del usuario es ambigua, o no se ajusta con claridad a ninguna categoría.

### 7. 4. 1. Implementación técnica

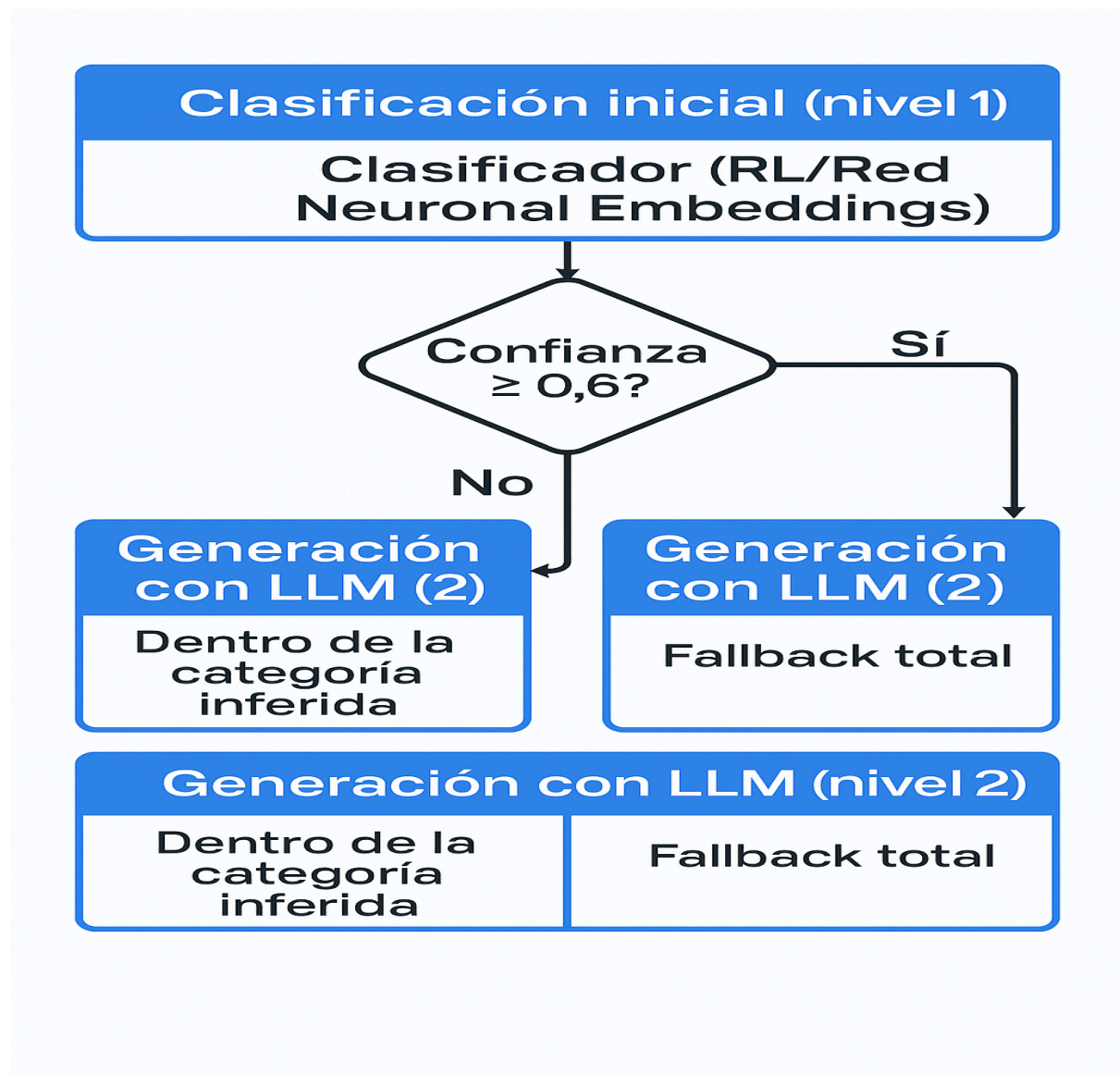
La función **run\_chatbot\_llm\_direct()** gestiona el flujo conversacional en tiempo de ejecución. Internamente, evalúa la confianza del clasificador y delega la generación al modelo LLM a través de la función **nodo\_recomendar\_destino()**.



Esta lógica ofrece la posibilidad de combinar la precisión del clasificador estructurado con la flexibilidad del modelo generativo.

**Figura 2.** Flujo de decisión híbrido de **CanBot**: integración entre clasificación ML/Red Neuronal con embeddings y generación con LLM.

El sistema evalúa la entrada del usuario mediante un clasificador de regresión logística o una red neuronal con embeddings según los requisitos de empleo. Si la confianza supera el umbral definido (0.6), se activa la generación contextual dentro de la categoría inferida. En caso contrario, el modelo LLM se encarga directamente de interpretar la intención y generar la recomendación.



## 8. Conclusiones

**CanBot** ofrece una experiencia de usuario flexible, eficiente y precisa gracias al uso de un clasificador de regresión logística / Red Neuronal con embeddings dinámicos en el primer nivel, así como a la estructura equilibrada del árbol de decisión y del corpus, que minimiza sesgos y simplifica la ampliación a nuevas categorías.

Su arquitectura modular, híbrida y recursiva posibilita la integración con modelos más complejos, como el LLM (que solo se activa en el primer nivel cuando es realmente necesario), así como la incorporación sencilla de mejoras posteriores.

Si en el futuro se desarrollan modelos más precisos o ligeros para clasificación de intenciones en lenguaje natural, estos podrán sustituir cualquiera de los tres modelos sin necesidad de reestructurar el sistema completo.

## 9. Anexos

Se adjunta, a continuación, el siguiente enlace: [Implementación de CanBot](#) con la documentación necesaria para su implementación, así como con las técnicas de visualización de los datasets, explicaciones, pruebas y evaluaciones realizadas durante el proceso.

## Fase 5: Evaluación y Conclusiones

### 5. a) Evaluación del sistema CanBot

El sistema **CanBot** ha sido sometido a pruebas y evaluaciones para verificar su correcto funcionamiento. El procedimiento y los resultados de las mismas se encuentran descritos en el enlace, [Implementación de CanBot](#).

Además, de las métricas cuantitativas de evaluación a las que han sido sometidos de manera independiente los modelos que lo integran, **CanBot** se ha evaluado cualitativamente con óptimos resultados.

Asimismo, el sistema final, con la integración de sus módulos independientes, se ha evaluado mediante una función, diseñada en los dos niveles jerárquicos descritos, a través de un conjunto de prueba compuesto por quince frases que simulan intenciones del usuario expresadas en lenguaje natural.

Los resultados se han presentado en tres métricas complementarias:

**a) Precisión del clasificador (nivel 1):** porcentaje de frases correctamente clasificadas por el modelo elegido (RL o SBERT ajustado).

**b) Recuperación por LLM (nivel 2):** porcentaje de frases incorrectamente clasificadas en el nivel 1, pero correctamente resueltas por el LLM.

**c) Precisión global combinada:** porcentaje total de aciertos sumando ambos niveles.

Esta evaluación final ha reflejado un rendimiento excelente, con una precisión del 100 % en el primer nivel jerárquico, que demuestra la eficiencia del sistema **CanBot**, ya que ha evitado que el LLM haya sido requerido en esa capa en todos los casos probados.

## 1. Ventajas del sistema CanBot

### 1.1. Arquitectura híbrida, modular y recursiva

Una de las principales fortalezas de **CanBot** radica en su arquitectura jerárquica, modular y recursiva, articulada en todas las fases de su diseño e implementación.

Desde la estructura simétrica del árbol de decisión, que facilita una navegación lógica y balanceada, hasta la organización del corpus sintético y el entrenamiento de los clasificadores, el sistema se ha concebido como un conjunto de módulos independientes, flexibles y fácilmente intercambiables.

Esta modularidad permite combinar distintas técnicas de procesamiento del lenguaje natural, tanto propias del Machine Learning como del Deep Learning:

- a) **Regresión logística (RL)**, que destaca por su bajo coste computacional y eficiencia cuando las entradas están bien definidas.
- b) **Red neuronal con embeddings contextuales** derivados de arquitecturas Transformer encoder (como SBERT), que superan las limitaciones de los modelos lineales, gracias a su capacidad para transformar el espacio semántico y resolver ambigüedades, como demuestra su precisión macro superior al 94%;
- c) **Modelo generativo del lenguaje (LLM)** basado en una arquitectura Transformer decoder, específicamente ajustada (Meta-Llama 3.1-8B-Instruct), que genera respuestas coherentes ante entradas abiertas.

Además, la hibridez del sistema ofrece la posibilidad de seleccionar la técnica más adecuada en función del tipo de entrada, según se priorice la eficiencia (RL), la profundidad semántica (embeddings) o la fluidez conversacional (LLM) y optimiza el rendimiento.

Finalmente, otra ventaja del sistema es la recursividad, tanto estructuralmente, mediante el árbol de decisión jerárquico, como a través de la función que vertebra el chatbot **CanBot**.

## 2) Limitaciones del sistema

Entre las principales limitaciones de **CanBot** se hallan:

### a) Los datasets

#### a) 1. Carácter sintético

A pesar de que los corpus aportados han sido diseñados con un equilibrio léxico y temático, su carácter sintético podría limitar la capacidad del sistema para generalizar frente a expresiones empleadas por usuarios reales.

#### a) 2. Tamaño y variabilidad semántica

Algunas categorías presentan solapamientos semánticos que afectan negativamente el rendimiento de los modelos lineales, como se refleja en los valores más bajos del

F1-score en determinadas clases o en las visualizaciones realizadas mediante técnicas como PCA (Principal Component Analysis).

Si bien este aspecto se puede mitigar con los embeddings contextuales, es una debilidad inherente a la estructura de los datos que debe mejorarse con la generación de datos sintéticos, parafraseos, sustitución de sinónimos o reordenamiento de frases.

#### **b) Imprecisión o errores en las respuestas**

Aunque en las evaluaciones cualitativas desarrolladas se ha verificado que las respuestas aportadas por **CanBot** son acertadas, ocasionalmente, puede suceder que, si el modelo LLM no conoce la respuesta, dé una alternativa errada.

Para evitar esta previsible falta, que caracteriza a los grandes modelos del lenguaje, se propone mejorar el sistema con técnicas de generación aumentada por recuperación RAG (Retrieval-Augmented Generation), *grounding* documental e ingeniería de *prompts*.

#### **c) Coste ambiental del modelo LLM**

A pesar de su activación controlada en **CanBot**, el empleo de un modelo LLM conlleva un alto gasto energético, debido al número de parámetros necesarios para su entrenamiento.

Este hecho plantea la necesidad de explorar alternativas, como la creación de algoritmos más sostenibles o la incorporación del modelo basado en SBERT en el segundo nivel jerárquico, en cuyo caso debiera optimizarse su rendimiento.

#### **d) Limitación en la evaluación de las respuestas generadas por CanBot**

Una de las principales limitaciones de la evaluación del sistema **CanBot** radica en su simplicidad que subestima la capacidad real del modelo generativo para reconocer intenciones expresadas mediante reformulaciones.

Se propone, por ello, como mejora futura, incorporar evaluaciones basadas en modelos de similitud semántica como Sentence Transformer.

## 5. b) Conclusión

**CanBot** representa una solución híbrida y éticamente consciente, que equilibra precisión y eficiencia mediante la integración jerárquica de un clasificador tradicional, una red neuronal basada en embeddings y un modelo LLM.

El modelo de lenguaje afinado (Meta-Llama 3.1-8B-Instruct), aporta a **CanBot** un valor significativo, dado que está preentrenado sobre un corpus muy superior en volumen y diversidad lingüística al empleado en el entrenamiento base de la versión ajustada del modelo SBERT (centrado en tareas de similitud semántica) y al de la regresión logística, entrenada únicamente sobre el corpus sintético diseñado para el primer nivel.

De este modo, a pesar de que en la evaluación final del sistema **CanBot** no ha sido necesaria su activación en el primer nivel jerárquico, los ensayos previos han demostrado que el LLM es capaz de recuperar frases mal clasificadas por los otros modelos, debido a las limitaciones del corpus o cuando las entradas del usuario se salen del dominio esperado.

Esta diferencia en la magnitud de datos previos, unida a su arquitectura Transformer decoder, explica, además, la capacidad del LLM para generar respuestas fluidas que propician una interacción natural con el usuario.

En consecuencia, sin su incorporación, la precisión de **CanBot** sería menor (como se demuestra en la documentación sobre la evaluación del modelo SBERT para 20 clases) y la interacción con el usuario forzada, en el marco estructurado del árbol de decisión.

Sin embargo, su integración en el sistema no pretende sustituir a los clasificadores clásicos o basados en *embeddings*, sino complementarlos en situaciones de baja confianza, pues desde **CanBot** somos conscientes de que el principal reto asociado a los LLMs es su sostenibilidad ambiental, desafío al que confiamos se dé pronta respuesta mediante el desarrollo de algoritmos de menor complejidad computacional que permitan diseñar grandes modelos del lenguaje más ligeros y eficientes.