**Manual de Usuario: Sistema de Clasificación de Tickets de Soporte**

**Tabla de Contenidos**

1. [Introducción](#introducci%C3%B3n)
2. [Requisitos del Sistema](#requisitos-del-sistema)
3. [Descripción de los Scripts](#descripci%C3%B3n-de-los-scripts)
4. [Configuración del Entorno](#configuraci%C3%B3n-del-entorno)
5. [Proceso Paso a Paso](#proceso-paso-a-paso)
   * [Limpieza de Datos](#1-limpieza-de-datos-limpiezapy)
   * [Creación y Entrenamiento del Modelo](#2-creaci%C3%B3n-y-entrenamiento-del-mod)
   * [Mejora del Modelo](#3-mejora-del-modelo-mejorarpy)
   * [Clasificación de Nuevos Tickets](#4-clasificaci%C3%B3n-de-nuevos-tickets-)
6. [Interpretación de Resultados](#interpretaci%C3%B3n-de-resultados)
7. [Resolución de Problemas](#resoluci%C3%B3n-de-problemas)
8. [Glosario](#glosario)

**Introducción**

Este sistema de clasificación de tickets de soporte utiliza técnicas de aprendizaje profundo para categorizar automáticamente los problemas reportados por los usuarios y asignarles una prioridad adecuada. El objetivo es mejorar la eficiencia en la gestión de tickets de soporte, permitiendo una respuesta más rápida y consistente a los problemas de los usuarios.

**Beneficios principales**:

* Automatización de la clasificación de tickets
* Mejora en los tiempos de respuesta
* Consistencia en la asignación de categorías y prioridades
* Facilitación del análisis de tendencias en los problemas reportados

**Requisitos del Sistema**

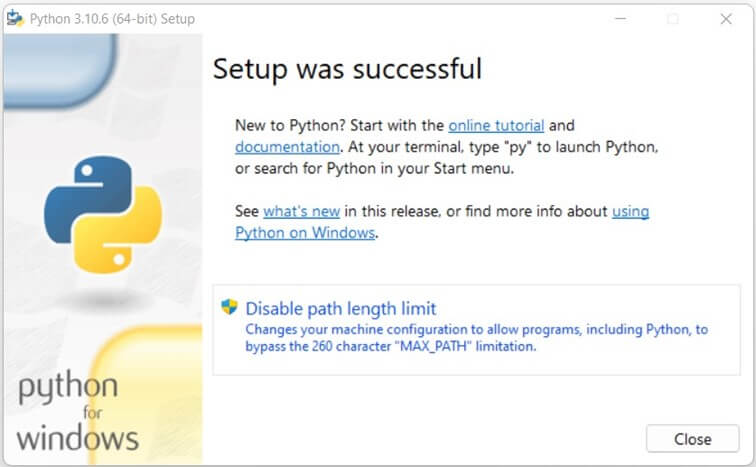
* Python 3.8 o superior
* Bibliotecas: pandas, numpy, tensorflow, scikit-learn, pickle5
* Archivos de datos:
  + contactos2024.csv: Datos iniciales para limpiar
  + etiquetadoManual.csv: Datos etiquetados manualmente para mejora del modelo
  + contactos2024\_limpio.csv: Nuevos tickets con datos limpios para clasificar
  + modelo\_ticketing.keras: Primer modelo entrenado
  + modelo\_ticketing\_ajustado.keras: Modelo re-entrenado a partir de el dataset etiquetado manualmente

**Descripción de los Scripts**

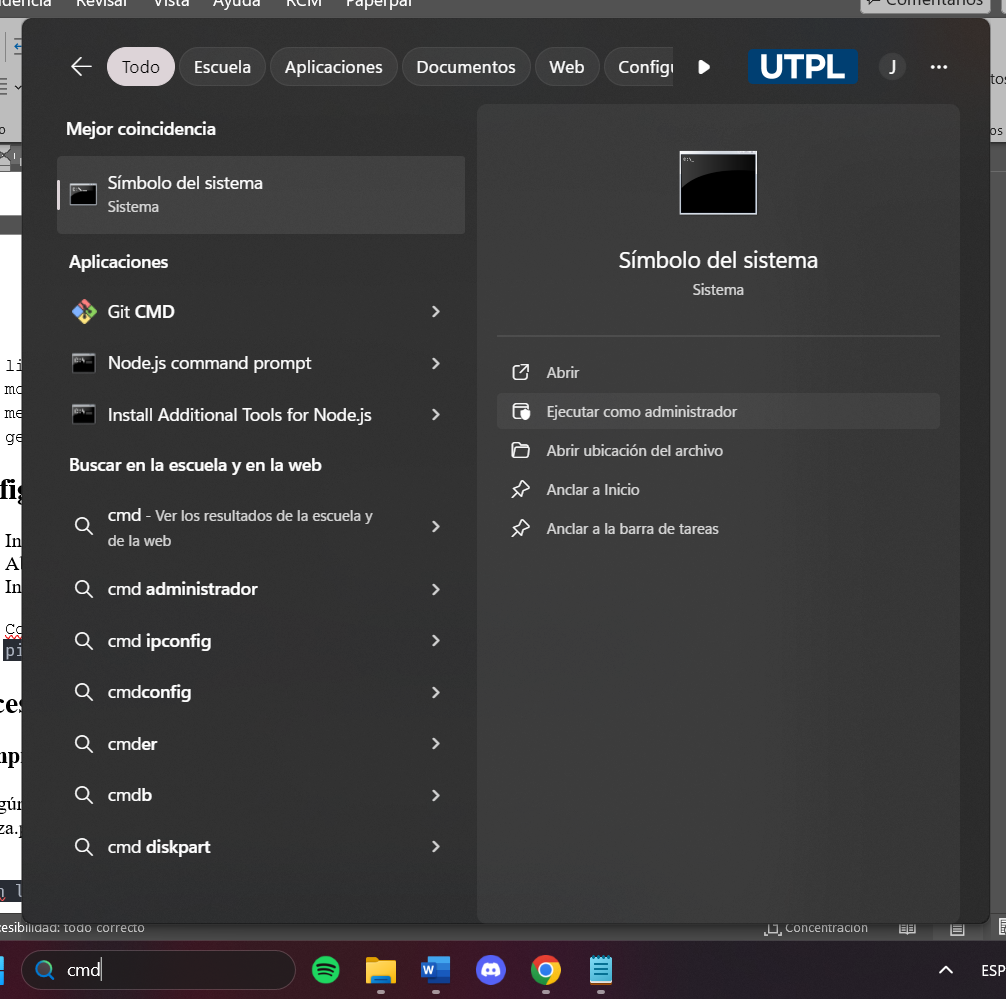
1. limpieza.py: Prepara y limpia los datos iniciales.
2. modelo.py: Crea y entrena el modelo de clasificación inicial.
3. mejorar.py: Mejora el modelo con datos adicionales etiquetados manualmente.
4. generacion.py: Utiliza un modelo entrenado para clasificar nuevos tickets.

**Configuración del Entorno**

1. Instala Python 3.8 o superior en tu sistema.



1. Abre una terminal o línea de comandos y preferiblemente ejecutala como administrador.



1. Instala las bibliotecas necesarias:

Copy

pip install pandas numpy tensorflow scikit-learn pickle5

**Proceso Paso a Paso**

**1. Limpieza de Datos (limpieza.py)**

a) Asegúrate de que el archivo 'contactos2024.csv' esté en la misma carpeta que limpieza.py. b) Ejecuta el script:

Copy

python limpieza.py

Este script realiza las siguientes operaciones:

* Carga los datos del archivo CSV.
* Elimina columnas innecesarias.
* Limpia el texto de los mensajes.
* Asigna calificaciones basadas en el contenido del mensaje.
* Convierte todo el texto a minúsculas.
* Guarda los datos limpios en 'contactos2024\_limpio.csv'.

**2. Creación y Entrenamiento del Modelo (modelo.py)**

a) Verifica que 'contactos2024\_limpio.csv' esté en la misma carpeta.

b) Ejecuta el script:

Copy

python modelo.py

Este script realiza lo siguiente:

* Carga los datos limpios.
* Clasifica los mensajes en categorías predefinidas.
* Prepara los datos para el entrenamiento (tokenización y secuenciación).
* Crea y entrena un modelo de red neuronal LSTM.
* Evalúa el modelo.
* Guarda el modelo entrenado, el tokenizador y las clases de etiquetas.

**3. Mejora del Modelo (mejorar.py)**

a) Asegúrate de tener un archivo 'etiquetadoManual.csv' con datos etiquetados manualmente.

b) Ejecuta el script:

Copy

python mejorar.py

Este script:

* Carga el modelo previamente entrenado.
* Prepara los datos etiquetados manualmente.
* Realiza un fine-tuning del modelo con estos nuevos datos.
* Guarda el modelo mejorado.

**4. Clasificación de Nuevos Tickets (generacion.py)**

a) Asegúrate de tener un archivo 'contactos2024\_limpio.csv' con nuevos tickets para clasificar.

b) Ejecuta el script:

Copy

python generacion.py

Este script:

* Carga el modelo mejorado.
* Prepara los nuevos datos para la clasificación.
* Clasifica los nuevos tickets.
* Asigna prioridades basadas en las categorías predichas.
* Guarda los resultados en 'contactos2024\_clasificado.csv'.

**Interpretación de Resultados**

El archivo 'contactos2024\_clasificado.csv' contendrá las siguientes columnas adicionales:

* categoria\_predicha: La categoría asignada por el modelo.
* prioridad\_predicha: La prioridad asignada basada en la categoría.

Las categorías y prioridades se asignan de la siguiente manera:

* Problemas con el pago: Prioridad Inmediata
* Problemas con el viaje: Prioridad Alta
* Problemas de seguridad: Prioridad Inmediata
* Problemas con la cuenta/app: Prioridad Alta
* Problemas de comportamiento: Prioridad Urgente
* Mensaje de operador: Prioridad Baja
* Otros problemas: Prioridad Baja
* Objetos perdidos: Prioridad Urgente
* Vocabulario inadecuado: Prioridad Alta

**Resolución de Problemas**

1. **Error de biblioteca faltante**: Asegúrate de haber instalado todas las bibliotecas necesarias utilizando el comando pip proporcionado.
2. **Archivo no encontrado**: Verifica que todos los archivos de datos estén en la misma carpeta que los scripts correspondientes.
3. **Error de memoria**: Si el sistema se queda sin memoria, considera procesar los datos en lotes más pequeños o utilizar una máquina con más RAM.
4. **Baja precisión del modelo**: Considera reentrenar el modelo con más datos o ajustar los hiperparámetros en modelo.py (por ejemplo, aumentar el número de épocas o cambiar la arquitectura de la red).
5. **Clasificaciones incorrectas**: Revisa las categorías predefinidas y las palabras clave asociadas en modelo.py. Puede ser necesario ajustarlas basándote en los resultados observados.

**Glosario**

* **CSV (Comma-Separated Values)**: Formato de archivo que almacena datos tabulares en texto plano.
* **Tokenización**: Proceso de dividir el texto en unidades más pequeñas llamadas tokens.
* **LSTM (Long Short-Term Memory)**: Tipo de red neuronal recurrente capaz de aprender dependencias a largo plazo en datos secuenciales.
* **Fine-tuning**: Proceso de ajuste fino de un modelo previamente entrenado con nuevos datos.
* **Hiperparámetros**: Parámetros del modelo que se establecen antes del entrenamiento y que afectan el proceso de aprendizaje.