Daniela Echavarria Yepes – 202111348

Juan Manuel Rodriguez – 202013372

Diego Alejandro Molano Roa – 202123015

Inteligencia de negocios

Proyecto 1 Etapa 2

Sección 1. (20%) Aumentación de datos y reentrenamiento del modelo.

1.1 Evaluación del modelo previo (v1) en el nuevo conjunto.

El modelo Logistic Regression (v1), el más efectivo de la etapa 1, se empleó en esta fase para examinar su desempeño en el nuevo conjunto de datos Datos_Etapa2.xlsx.

El pipeline de preprocesamiento permaneció igual al de la fase anterior, con limpieza, lematización utilizando spaCy y vectorización TF-IDF (con un rango de ngramas de 1 a 2, un mínimo de documentos con frecuencia (ngram_range = (1, 2), min_df = 3, max_df = 0.9).

Se almacenaron en los archivos los resultados conseguidos en este nuevo conjunto:

- baseline_etapa2_macro.csv
- baseline_etapa2_por_clase.csv
- predicciones_v1_en_nuevo_con_score.csv

El modelo obtuvo un valor de Accuracy de 0.9608 y un F1-macro de 0.9559, lo cual demuestra un rendimiento estable; sin embargo, el recall en la clase ODS 1 (Fin de la pobreza) fue inferior, aunque esta sigue siendo la más desbalanceada.

1.2 Aumentación de datos mediante prompting

Se llevó a cabo una estrategia de data augmentation con prompting para reducir el desequilibrio.

Se utilizó la API de OpenAI (modelo gpt-4o-mini) con indicaciones particulares para cada clase:

- ODS 1: pobreza, informalidad, brechas rurales y vulnerabilidad económica.
- ODS 3: salud pública, salud mental, EPS/IPS, prevención y acceso a medicamentos.
- ODS 4: educación, cobertura, calidad docente e infraestructura.

El generador retornó un array JSON formado por párrafos en español (de 3 a 5 oraciones), que después fue validado y depurado.

Se eliminaron textos vacíos y duplicados, y se filtraron aquellos que no se ajustaban al rango de tokens previsto (de 8 a 200).

Por último, los textos sintéticos fueron incorporados en data/synthetic_multiclase_clean.csv, elevando las clases minoritarias hasta que su tamaño se aproxime al 90 % del de la clase mayoritaria.

1.3 Reentrenamiento del modelo (v2)

Los tres conjuntos se unieron:

- Datos_Etapa2.xlsx + sintéticos_multiclase + Datos_proyecto.xlsx (original)

Para asegurar la comparabilidad, el modelo de regresión logística (v2) fue reentrenado con los mismos hiperparámetros y configuración TF-IDF.

El nuevo modelo se evaluó en el mismo conjunto de prueba fijo (*hold-out Etapa 1*), obteniendo:

Modelo	Accuracy	F1-macro	F1 ODS1	F1 ODS3	F1 ODS4
v1 (sin aumentación)	0.9608	0.9559	0.9333	0.9584	0.9758
v2 (con aumentación)	0.9649	0.9617	0.9458	0.9636	0.9756
Δ (v2 − v1)	+0.0041	+0.0058	+0.0125	+0.0051	−0.0002

Los resultados se almacenaron en:

- figures/tabla_comparativa_v1_vs_v2_mismo_test_CLEAN.csv
- figures/f1_por_clase_mismo_test.png.

1.4 Evaluación en el nuevo set (opcional)

Para comprobar su generalización, el modelo v2 también se probó en Datos_Etapa2.xlsx:

Se logró un F1-macro ≈ 0.96 y una precisión de aproximadamente 0.965, con un evidente avance en la clase ODS 1.

Los resultados se encuentran en las siguientes ubicaciones:

- outputs/predicciones_v2_en_nuevo_set_con_score.csv
- outputs/nuevo_set_v2_macro.csv.

donde también se incluye el puntaje (la probabilidad vinculada a la predicción), lo que cumple con los requisitos de la rúbrica.

1.5 Conclusión

El procedimiento de aumentación mediante prompting multiclase posibilitó que las clases minoritarias (ODS 1 y 3) contaran con una mejor representación, lo cual a su vez hizo posible que las métricas globales del modelo se incrementaran.

El F1-score de ODS 1 se incrementó en +0.0125, lo que muestra un balance más favorable entre la precisión y el recall.

Dado que el modelo v2 preserva la misma arquitectura y los mismos parámetros, se puede afirmar que las mejoras son consecuencia directa de la estrategia de aumentación de datos.

Para resumir, la metodología de prompting guiado fue efectiva para crear datos sintéticos realistas y equilibrar el conjunto de entrenamiento, logrando así las metas establecidas en la Sección 1 del proyecto.

Sección 2. (15%) Proceso de automatización del proceso de preparación de datos, construcción del modelo, persistencia del modelo y acceso por medio de API:

Con el objetivo de garantizar la reproducibilidad y escalabilidad del flujo de aprendizaje automático, se desarrolló un proceso automatizado que integra la preparación de datos, la construcción del modelo, la persistencia de artefactos y el acceso mediante API REST. Esto permite que el modelo de clasificación sea reutilizado de manera modular en producción.

2.1 Automatización y construcción del pipeline

Utilizando scikit-learn se implementó un pipeline que unifica todas las etapas necesarias para el entrenamiento y la inferencia del modelo. El pipeline incluye:

1. Vectorización TF-IDF: parámetros ajustados para minimizar ruido (min_df=3, max_df=0.9, sublinear_tf=True, norm="l2").
2. Clasificador Logistic Regression: mejorado con balance automático de clases (class_weight="balanced") y un número elevado de iteraciones para asegurar convergencia (max_iter=2000).

2.2 Persistencia del modelo

Para mantener la trazabilidad y disponibilidad del modelo entrenado se implementó un sistema de persistencia utilizando la librería joblib. Con este se generan artefactos para la clasificación y vectorización, almacenados en formato binario en la carpeta /models.

Las funciones save_model() y load_current_model() permiten guardar y cargar estos archivos, asegurando que cualquier ejecución de la API utilice la versión más reciente del modelo. Con cada reentrenamiento los artefactos son reemplazados, manteniendo la coherencia entre los vectores y el modelo.

2.3 API REST para predicción y reentrenamiento

Con el objetivo de exponer el modelo como un servicio reutilizable, se desarrolló una API RESTful utilizando el framework FastAPI.

La API define tres endpoints principales:

/health – Verificación del estado

Permite comprobar la disponibilidad del servicio y la correcta carga del modelo, antes de realizar operaciones POST.

Devuelve un JSON con el estado (ok / error) y detalles del modelo cargado, facilitando el monitoreo durante el despliegue.

/predict – Predicción de nuevas instancias

Recibe una lista de textos en formato JSON y devuelve sus categorías predichas (ODS 1, 3 o 4) en el mismo orden. El proceso consiste en cargar el modelo, vectorizarlo, realizar las transformaciones TF-IDF, generar las predicciones y almacenarlas como una lista JSON.

Ejemplo de solicitud:

```
{
  "textos": [
    "La educación rural necesita más inversión",
    "El acceso a la salud pública ha mejorado"
  ]
}
```

Respuesta:

```
{
  "predicciones": [4, 3]
}
```

/retrain – Reentrenamiento con nuevos datos

Recibe nuevas instancias junto con sus etiquetas y actualiza el modelo actual, reemplazando la versión anterior. Durante este proceso inicialmente, se intenta cargar el modelo actual, si no existe se construye uno nuevo con build_pipeline(). Los textos generados se vectorizan y el modelo se ajusta con fit(), calculando sus métricas de rendimiento. Finalmente se almacena el modelo y se devuelve un reporte en formato JSON. Se utilizó Pydantic para la validación de datos de entrada.

Ejemplo de solicitud:


```
{
  "textos": [
    "La pobreza infantil sigue siendo un reto nacional",
    "Los hospitales rurales necesitan más personal médico"
  ],
  "labels": [1, 3]
}
```


Respuesta:


```
{
  "mensaje": "Modelo reentrenado y actualizado",
  "metrics": {
    "accuracy": 0.965,
    "f1_macro": 0.958,
    "recall_macro": 0.955,
    "precision_macro": 0.960
  }
}
```


2.4 Definiciones y estrategias de reentrenamiento

Se exploraron tres enfoques de reentrenamiento valorando su facilidad de aplicación y calidad de resultados al problema de clasificación de textos.

Tipo	Descripción	Ventaja	Desventaja
Completo	Reentrena el modelo desde cero con todos los datos disponibles (originales, nuevos y sintéticos).	Elimina sesgos acumulados y aprovecha el contexto completo.	Requiere más tiempo y recursos computacionales.
Incremental	Ajusta los pesos del modelo existente con los nuevos datos, preservando el	Rápido y eficiente; permite aprendizaje continuo.	Puede propagar errores o sesgos del modelo original.

	conocimiento previo.		
Batch periódico	Reentrena el modelo de forma programada con bloques de datos acumulados.	Controla la frecuencia de actualización, aportando estabilidad.	Datos recientes se incorporan con retardos.

En esta etapa se implementó el reentrenamiento completo, dado que el tamaño de los datasets y los recursos disponibles permiten recalcular el modelo de manera integral sin afectar la eficiencia.

Esta elección asegura que las mejoras obtenidas por aumentación y nuevos datos se integren en una única versión coherente.

2.5 Despliegue e integración

El backend fue configurado con FastAPI para ejecutarse con el comando: uvicorn app:app --reload --port 8000

El servicio expone los endpoints API REST en localhost:8000, permitiendo integrarse directamente con el frontend desarrollado en React.

Se definió un middleware para permitir la comunicación entre ambos ambientes de desarrollo

Dado el objetivo de asegurar escalabilidad por contenerización se definió una clara estructura de carpetas para almacenar todos los artefactos, gráficas, modelos y csv generados durante el entrenamiento y despliegue.

2.6 Conclusión

La automatización del proceso de modelado por clasificación y el desarrollo de la API REST con FastAPI definen un flujo reproducible, escalable y orientado a una fácil integración.

Con esto, los analistas pueden interactuar directamente con el modelo para generar predicciones o incorporar nuevos datos, sin conocimiento técnicos de modelado o procesamiento de datos.

Sección 3. (25%) Desarrollo de la aplicación y justificación.

3.1 Descripción del usuario/rol de la organización que va a utilizar la aplicación, conexión con el proceso de negocio y relevancia

Usuario o rol principal

El **Fondo de Población de las Naciones Unidas (UNFPA)** y las entidades aliadas de planeación territorial son los principales usuarios institucionales de la

aplicación.
Dentro de la organización, el rol directamente beneficiado es el de analista de políticas públicas y participación ciudadana, responsable de recopilar, depurar y clasificar los aportes ciudadanos provenientes de talleres, foros o plataformas virtuales de consulta sobre los Objetivos de Desarrollo Sostenible (ODS).

Conexión con el proceso de negocio

El proceso institucional al que se vincula la aplicación es el de seguimiento y evaluación participativa de los ODS, específicamente la etapa de análisis cualitativo de percepciones ciudadanas.
Actualmente, este proceso depende de la lectura manual de miles de comentarios y encuestas abiertas.
La nueva aplicación automatiza la clasificación semántica de textos en función de los ODS 1 (Fin de la pobreza), ODS 3 (Salud y bienestar) y ODS 4 (Educación de calidad).
De esta forma:

1. **Reduce tiempos** de análisis y permite procesar grandes volúmenes de información en minutos.
2. **Estandariza criterios** de codificación, evitando sesgos individuales.
3. **Alimenta los tableros de monitoreo** institucionales con indicadores derivados del discurso ciudadano.
4. **Cierra el ciclo de retroalimentación**: los analistas pueden re-entrenar el modelo con nuevos datos para mejorar su precisión y adaptarlo a contextos locales.

Importancia para el rol y para la organización

Para los analistas y planificadores, la existencia de esta aplicación es **estratégica** porque:

- Permite **tomar decisiones basadas en evidencia** textual y no solo en indicadores cuantitativos.
- Fortalece la **transparencia y trazabilidad** del proceso participativo, demostrando cómo la voz de la ciudadanía influye en las políticas públicas.
- Facilita la **priorización de territorios y problemáticas**, orientando recursos hacia los grupos más vulnerables.
- Promueve la **sostenibilidad del conocimiento institucional**, ya que el modelo aprende progresivamente del trabajo humano.

En síntesis, la aplicación integra analítica de texto, inteligencia artificial y participación ciudadana, transformando un proceso tradicionalmente manual y

fragmentado en un sistema **inteligente, replicable y alineado con la Agenda 2030**.

¿Qué recursos informáticos requiere para entrenar, ejecutar, persistir el modelo analítico y desplegar la aplicación?

El entrenamiento, ejecución, persistencia y despliegue del modelo analítico requieren recursos informáticos de nivel medio, suficientes para procesar textos en lenguaje natural de manera eficiente. El modelo se persiste en formato .joblib dentro de un entorno local o en la nube, junto con archivos de métricas en formato CSV. La API, desarrollada en FastAPI y desplegada mediante uvicorn, expone los endpoints /predict y /retrain, mientras que la aplicación frontend en React (con Vite) se ejecuta en Node.js y se comunica con el backend mediante solicitudes JSON, aseguradas con políticas CORS.

¿Cómo se integrará la aplicación construida a la organización, estará conectada con algún proceso del negocio o cómo se pondrá a disposición del usuario final?

La aplicación se integrará como un servicio interno de analítica dentro del flujo de planificación participativa y monitoreo de políticas públicas: recibirá textos desde los canales actuales de recolección (formularios web, encuestas, buzones ciudadanos, CRM o mesa de ayuda) mediante un conector o una cola (p. ej., webhook/Batch → API /predict), devolverá la categoría ODS y su probabilidad; el reentrenamiento (/retrain) se ejecutará de forma controlada por analistas o como job programado con nuevos corpus validados. El acceso del usuario final será a través de una interfaz web React (SSO/roles: analista, planificador, supervisor),

¿Qué riesgos tiene para el usuario final usar la aplicación construida?

El principal riesgo para el usuario final radica en la interpretación automática de los textos, ya que el modelo podría clasificar de forma incorrecta una opinión o sesgar los resultados si el conjunto de entrenamiento no es representativo o contiene errores. Asimismo, existe riesgo de malinterpretación de los resultados por parte del usuario, especialmente si no comprende las métricas de confianza o las limitaciones del modelo. En materia de privacidad, el envío de textos con datos personales sin anonimización puede vulnerar la normativa de protección de datos.

Sección 4. (18%) Resultados.

El video se puede encontrar en el github llamado proyecto 1 – Etapa 2

Sección 5. (10%) Trabajo en equipo.

Roles

Líder de proyecto: Daniela

- Tareas:

1. Distribuir labores (1 hora)
2. Organizar timelines (1 hora)

- Confirmación de cumplimiento de objetivos (3 hora)
- Realizacion del front y del video (3 horas)

Líder de negocio: Diego

- Tareas:
- Implementación del pipeline y guardado del modelo (2 horas)
- Implementacion del api mediante endpoints (4 horas)

Líder de datos: Juan Manuel

- Tareas:

1. Reentrenamiento del modelo mediante el uso de tecnicas de prompting (4 horas)
2. Generación de archivos excel y csv de resultados (2 horas)

Líder de analítica: Daniela

- Tareas:

1. Validación de resultados (3 horas)
2. Documentación de los realizado (2 horas)

División de puntos

- Daniela: 34
- Diego: 32
- Juan Manuel: 34

Los tres integrantes demostraron entendimiento del problema y proactividad en el desarrollo de las soluciones.

Uso de IA

Durante el desarrollo del proyecto, la inteligencia artificial fue un recurso clave para aprender, explorar y resolver problemas técnicos de manera más ágil. En particular, la utilizamos en cuatro dimensiones:

- En este período, la inteligencia artificial fue empleada para ayudar a comprender y aplicar de manera adecuada el procedimiento de aumento de

datos mediante prompting y el reentrenamiento del modelo base (Regresión logística).

Ayudó a aclarar el flujo total del pipeline: desde la evaluación del modelo original (v1), pasando por la producción de textos sintéticos para las clases ODS a través de la API de OpenAI, hasta el desarrollo del nuevo modelo (v2) y su comparación cuantitativa.

- Apoyo en la depuración de código (debugging): cuando aparecieron errores de sintaxis, conflictos con librerías o dificultades en la vectorización de textos, la IA nos ayudó a identificar las causas y proponer soluciones concretas. Este acompañamiento fue clave para mantener un flujo de trabajo continuo y evitar bloqueos prolongados.
- La IA también fue utilizada directamente dentro del proyecto como fuente de generación de datos, a través de prompting multiclase para producir textos representativos de las clases ODS 1, ODS 3 y ODS 4.
  Esta técnica permitió aplicar aumentación de datos (data augmentation) de manera controlada y contextual, manteniendo la coherencia lingüística y temática de los textos.
- Guía metodológica y estructuración: la IA fue útil para proponer esquemas de análisis de resultados, resúmenes comprensibles para audiencias no técnicas y recomendaciones de cómo comunicar los hallazgos desde una perspectiva de negocio.