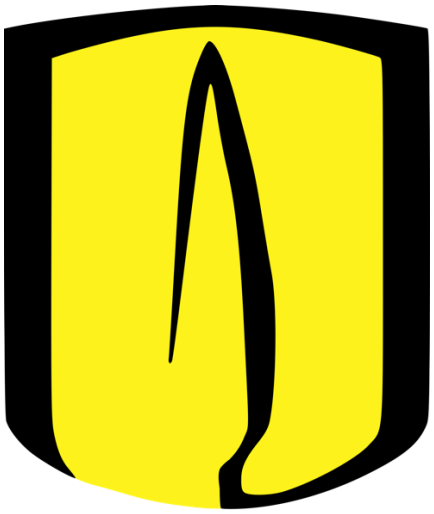
Proyecto 1 Etapa 2 – Analítica de textos

Evelin Vanessa Villamil Guerrero – 202113360
Carlos German Monroy Andrade – 201728260



Grupo G34

Universidad de los Andes
Ingeniería de Sistemas y Computación
Inteligencia de negocios

Tabla de contenidos

1. Creación del pipeline 2

2. Desarrollo de la aplicación 3

3. Utilidad de la aplicación en un negocio 4

4. Mejoras obtenidas con la ayuda del experto de estadística asignado 5

1. Creación del pipeline

Como primer paso para el desarrollo de nuestra aplicación, decidimos implementar un pipeline para poder estandarizar y automatizar el proceso de limpieza y transformación de los datos, junto con la predicción de sentimientos utilizando el modelo entrenado.

Para implementar el pipeline se tiene primero una clase "Limpieza" que hereda de dos clases: "BaseEstimator" y "TransformerMixin". Esta clase se utiliza para limpiar los datos de un DataFrame de Pandas que contiene opiniones o comentarios en español.

En el método "init" de la clase se inicializa un conjunto de stopwords en español que se utilizará más adelante para eliminar las palabras comunes que no aportan información relevante en el análisis. Por otro lado, en el método "fit" no se realiza ninguna tarea y simplemente devuelve el objeto "self", lo cual es una práctica común en Scikit-learn para mantener la coherencia del pipeline. Además, en el método "transform" se llama al método "preprocess", que es donde se realiza la limpieza de los datos. El método "preprocess" recibe un DataFrame de Pandas y realiza una serie de tareas de limpieza en el texto de cada comentario, que se detallan a continuación:

1. Remover los caracteres ASCII: utilizando la librería "unicodedata", se eliminan los caracteres especiales que no son parte del alfabeto español.
2. Cambiar las mayúsculas por minúsculas: se convierten todas las letras a minúsculas para homogeneizar el texto.
3. Eliminar los signos de puntuación: utilizando la librería "string", se eliminan los signos de puntuación del texto.
4. Reemplazar los números por su equivalente en palabras: utilizando la librería "num2words", se reemplazan los números que aparecen en el texto por su equivalente en palabras en español.

5. Eliminar las stopwords: utilizando la librería "nltk", se eliminan las palabras comunes que no aportan información relevante en el análisis.
6. Finalmente, el método "transform" devuelve el DataFrame limpio y preparado para ser utilizado en el siguiente paso del pipeline.

Por otro lado, en el archivo AnalísisTextosODS.ipynb de la etapa 1 se entrena y se exporta el pipeline como tal. Primero, se importa la clase "Limpieza" que se define en otro archivo y que se utiliza para limpiar los datos de texto. Luego, se define el pipeline que consta de tresetapas:

1. "preprocessor": la instancia de la clase "Limpieza" que se encarga de limpiar el texto.
2. "vectorizer": un vectorizador TF-IDF que convierte los comentarios limpios en vectores numéricos que se pueden utilizar como entrada para el modelo de clasificación.
3. "classifier": un clasificador RandomForestClassifier que se encarga de predecir la polaridad del sentimiento de los comentarios.

A continuación, se carga el conjunto de datos textos y se divide en conjuntos de entrenamiento y prueba utilizando la función "train_test_split". Luego, se ajusta el pipeline con los datos de entrenamiento mediante el método "fit" y se utiliza el método "predict" para hacer predicciones de sentimientos sobre los datos de entrenamiento y prueba. Finalmente, se guarda el joblib que contiene el pipeline entrenado.

2. Desarrollo de la aplicación

Para el back-end de la aplicación, usamos Python con el framework FastAPI. Este framework es conocido por su facilidad de uso, velocidad de ejecución y robustez. FastAPI utiliza el servidor web Uvicorn para ejecutar y exponer el API. La aplicación puede recibir un texto único en español. Luego, utiliza un modelo de aprendizaje automático previamente entrenado para predecir a que objetivo (6,7 o 16) pertenece una cadena de texto.

El primer endpoint make_prediction() también utiliza el método HTTP POST y recibe como entrada un objeto ReviewModel que representa una sola reseña en español. La reseña se transforma en un objeto pandas DataFrame, se utiliza el modelo para hacer la predicción y el resultado es un str con el número del objetivo.

En cuanto al front-end, usamos el framework Boostrap. Este es un popular framework de diseño web de código abierto que se utiliza para desarrollar sitios web y aplicaciones web de manera eficiente. Fue creado por Twitter y es mantenido por la comunidad de desarrolladores. Bootstrap proporciona un conjunto de herramientas, componentes y estilos predefinidos que facilitan la creación de interfaces web atractivas y responsivas.

La página puede cargar un solo texto y listar los resultados de los textos cargados. En la página de cargar una solo texto, hay un campo de texto donde el usuario puede escribir y, al hacer clic en "enviar texto", nos redirige a la ruta de listar, donde podemos ver el ODS predicho por el modelo. Por detrás, se envía el texto a la ruta que definimos en el back-end como "/predictone".

3. Utilidad de la aplicación en un negocio

- **Automatización de la Clasificación de Textos:** Desarrollar un modelo de clasificación de textos basado en técnicas de aprendizaje automático que permita asignar automáticamente un texto a uno de los Objetivos de Desarrollo Sostenible (ODS) específicos. Es importante porque agiliza el proceso de identificar los temas relacionados con los ODS en grandes volúmenes de datos, lo que ahorra tiempo y recursos.
- **Mejora de la Eficiencia en la Evaluación de Políticas Públicas:** Facilitar la identificación y evaluación de políticas públicas relacionadas con los ODS mediante el análisis automatizado de opiniones y comentarios de la población local. Es importante porque la mejora de la eficiencia en la evaluación de políticas públicas permite a UNFPA y otras entidades identificar desafíos y oportunidades más rápidamente, lo que a su vez acelera el proceso de toma de decisiones.
- **Identificación de Problemas y Soluciones de Manera Más Rápida y Precisa:** Proporcionar una herramienta que permita identificar problemas y soluciones en relación con los ODS en un contexto territorial de manera más rápida y precisa. Es importante porque la capacidad de identificar problemas y soluciones de manera eficiente ayuda a enfocar los esfuerzos en áreas críticas y a lograr un impacto más significativo en el desarrollo sostenible.

4. Mejoras obtenidas con la ayuda del experto de estadística asignado

El grupo de estadística nos hizo las siguientes recomendaciones sobre el entendimiento de datos y cómo podríamos mejorar en el análisis:

- **Paralelización del procesamiento de datos:** La paralelización del procesamiento es una manera de disminuir el tiempo necesario para la preparación y ejecución de los datos. Esto implica dividir la recopilación de datos en partes más pequeñas y procesarlas simultáneamente en varios núcleos o máquinas. Esto podría acortar significativamente el tiempo necesario

para preparar y ejecutar los datos.

- **Utilización de técnicas de procesamiento de datos en tiempo real:** en lugar de procesar todos los datos a la vez, se pueden utilizar técnicas de procesamiento de datos en tiempo real para procesar los datos a medida que llegan. Esto podría acortar el tiempo de preparación y la complejidad de los datos porque elimina la necesidad de esperar a que todos los datos estén disponibles antes de comenzar el procesamiento.
- **Optimización de algoritmos y operaciones:** Para acortar los tiempos de ejecución, es posible optimizar los algoritmos y operaciones utilizadas en la preparación de datos. Esto puede implicar, entre otras cosas, el uso de algoritmos más efectivos, la optimización de canalizaciones y operaciones vectorizadas.
- **Uso de herramientas y bibliotecas efectivas:** es fundamental utilizar herramientas y bibliotecas efectivas para el procesamiento de datos. Esto puede implicar el uso de bibliotecas de procesamiento de texto optimizadas para la tarea, como spaCy o fastText, que pueden acortar el tiempo necesario para completar las operaciones de limpieza y transformación de texto.
- **Adoptar técnicas eficientes de almacenamiento y acceso**: por ejemplo, se pueden utilizar técnicas eficientes de almacenamiento y acceso para mejorar el tiempo de preparación de datos y mejorar el acceso al software. Esto incluye el uso de bases de datos optimizadas para procesar grandes volúmenes de datos, así como el uso de técnicas de indexación para acelerar la búsqueda y recuperación de datos.

Con estas recomendaciones pudimos mejorar la parte del análisis y entendimiento de datos, de tal forma que desarrollamos la página web con un modelo más preciso que el creado inicialmente.