MILESTONE 1

Rule-based Text Classification

**Objetivo**

Desarrollar un sistema que clasifique comentarios en **ofensivos (1)** o **no ofensivos (0)** **utilizando reglas lingüísticas** extraídas del texto, sin usar aprendizaje automático todavía.

Para desarrollar correctamente el **Milestone 1 del proyecto HateBR**, nos basaremos principalmente en las **Prácticas 1 y 2.**

* La **Práctica 1** te da las herramientas de limpieza y análisis exploratorio.
* La **Práctica 2** te da las herramientas para extraer conocimiento lingüístico (gramática, sentimiento, entidades) con las que construirás tus reglas.

CONTEXTO:

HateBR es el primer conjunto de datos a gran escala, anotado por expertos, de comentarios brasileños en Instagram para la detección de lenguaje ofensivo y discurso de odio en la web y las redes sociales. El conjunto de datos se recopiló a partir de comentarios brasileños en Instagram dirigidos a políticos y fue anotado manualmente por especialistas. Comprende 7000 documentos anotados en tres capas diferentes: (i) clasificación binaria (comentarios ofensivos frente a no ofensivos), (ii) nivel de ofensividad (mensajes muy ofensivos, moderadamente ofensivos y ligeramente ofensivos) y (iii) objetivos del discurso de odio. Cada comentario fue anotado por tres expertos, lo que permitió alcanzar un alto grado de concordancia entre ellos.

A**ntes de limpiar y procesar el texto**, es **muy recomendable realizar una exploración inicial del dataset (EDA preliminar)** para comprender su naturaleza y **diseñar una estrategia de limpieza adecuada y no destructiva**.

**Orden de trabajo**

**Paso 1 – Exploración inicial**

🔹 Objetivo: conocer el contenido bruto del dataset.

Aquí **no limpiamos aún los textos**, solo los **observamos y describimos**:

* Cargamos el dataset.
* Comprobamos número de filas, columnas y valores nulos.
* Revisamos cuántas clases hay y cuántos ejemplos por clase.
* Mostramos algunas filas para entender el tipo de lenguaje (informal, con emojis, abreviaciones, etc.).

Esto nos permitirá decidir **qué tipo de limpieza es necesaria**.

**Paso 2 – Limpieza de datos (preprocesamiento básico)**

🔹 Objetivo: normalizar el texto antes del análisis lingüístico.

Este paso va **después de la exploración** y **antes de extraer tokens, n-gramas o entidades**.

Aquí tuvimos en cuenta varias cosas para una limpieza:

1. **Convertir a minúsculas** – para evitar duplicidad (“Burro” vs “burro”).
2. **Eliminar URLs, menciones, hashtags, emojis** (según tu criterio).
3. **Quitar signos de puntuación innecesarios** (excepto si te sirven como rasgos, por ejemplo exclamaciones).
4. **Eliminar caracteres no alfabéticos o espacios múltiples.**

**Después de analizar las salidas pertinentes, nos dimos cuenta que debemos hacer lo siguiente:**

1. **Mantener las mayúsculas: pueden aportan información que se clasifique como ofensiva**
2. **Quitar signos de puntuación innecesarios (mantener ?): puede que aporte ironía o burla y que se clasifique como ofensivo**
3. **Eliminar caracteres no alfabéticos o espacios múltiples: limpieza básica**
4. **Mantener emojis, menciones y hashtags: aportan información para clasificar**

**Sin embargo, no vamos a eliminar todavía las stopwords ni haremos lematización porque vamos a usar análisis lingüístico posterior con spaCy, pues esa librería puede beneficiarse de ver el texto completo.**

**Paso 3 – Extracción de conocimiento lingüístico**

Una vez limpios los textos, aplicamos spaCy, nltk o CountVectorizer para:

* Tokenizar.
* Obtener n-gramas frecuentes.
* Detectar entidades y POS tags.
* Calcular sentimientos o polaridad.

En tu caso concreto (**análisis de comentarios en portugués y detección de discurso ofensivo**), **sí es recomendable eliminar acentos**, por tres motivos:

1. **Homogeneidad del texto**  
   Muchos usuarios en redes sociales no escriben correctamente los acentos (por ejemplo, escriben *“nao”* en lugar de *“não”*).  
   Si conservas los acentos, “não” y “nao” serán tratadas como palabras distintas, aunque semánticamente sean iguales.
2. **Simplicidad de las reglas lingüísticas**  
   Al eliminar acentos, las expresiones regulares o las reglas manuales (por ejemplo, para detectar insultos) funcionan mejor porque no hay que incluir variantes con o sin tilde.
3. **Compatibilidad con modelos posteriores**  
   Si en *Milestone 2* entrenas un modelo clásico de ML (SVM, Naive Bayes, etc.), quitar acentos evitará que se duplique el vocabulario y ayudará a la generalización.

Paso 4 **– Creación de reglas**

**Una vez tenemos el dataset explorado, limpiado y con extracción lingüística relevante entonces planteamos las siguientes reglas:**

1. Si el post contiene algún insulto hecho 🡪 ofensivo
2. Si el post contiene algún insulto y está en MAYÚSCULAS 🡪 ofensivo (recuerda que todos los insultos están filtrados como minúsculas, entonces primero habría que detectar que la palabra está en mayúscula, pasarla a minúscula y comprobar si esta en la lista de insultos)
3. Si el post contiene algún insulto y signo de inerrogación “?” 🡪 ofensivo
4. Si el post completo supera un cierto umbral de polaridad 🡪 ofensivo (date cuenta que la polaridad se aplica sobre una oración completa, NO sirve igual si lo aplicamos palabra a palabra porque no tiene contexto)
5. Si el post contiene el patrón NOUN + ADJ 🡪 ofensivo (solo si NOUN o ADJ está dentro de nuestra lista de insultos)
6. Si el post contiene el patrón 2ª persona + ADJ negativo 🡪 ofensivo
7. Si el post contiene algún insulto y hay MAYÚSCULAS 🡪 ofensivo

NOTA: es importante diferenciar la lista de insultos (“cleaned\_insults”) 🡪 posibles palabras negativas que clasifican los posts como ofensivos y la lista de insultos hechos (“hybrid\_insults\_expanded”) 🡪 palabras propias portuguesas garantizadas como insultos hechos

SIGUIENTES PASOS 🡪 ANALIZAR MÁS PATRONES Y PONER MÁS REGLAS, LUEGO HAY QUE MARCAR EL ORDEN DE ACTIVACIÓN SEGÚN LA IMPORTANCIA Y POR ÚLTIMO IR PROBANDO LA MEJOR COMBINACION DE REGLAS Y COMPARAR MÉTRICAS PARA QUEDARNOS CON EL MEJOR CONJUNTO DE REGLAS PARA NUESTRO SISTEMA BASADO EN REGLAS