# Análisis de preguntas del INE

## 

## Programación para el análisis de datos

Rodolfo Luthe Narváez

Iñaki Sebastián Orozco García

Con la ayuda de Juan Jorge Mendoza García

# Introducción

Este proyecto busca procesar preguntas hechas a través de un formulario en línea al INE para ser preguntadas en debates presidenciales a los candidatos. El proyecto debe de implementar técnicas de procesamiento de texto para cerciorarse que las preguntas recopiladas sean apropiadas para presentarse en los debates, no pueden estar sesgadas o contener lenguaje impropio de ninguna índole.

Como resultado de este proyecto se deben entregar preguntas depuradas que no se repitan y cumplan con los criterios establecidos por el INE.

## Hallazgos

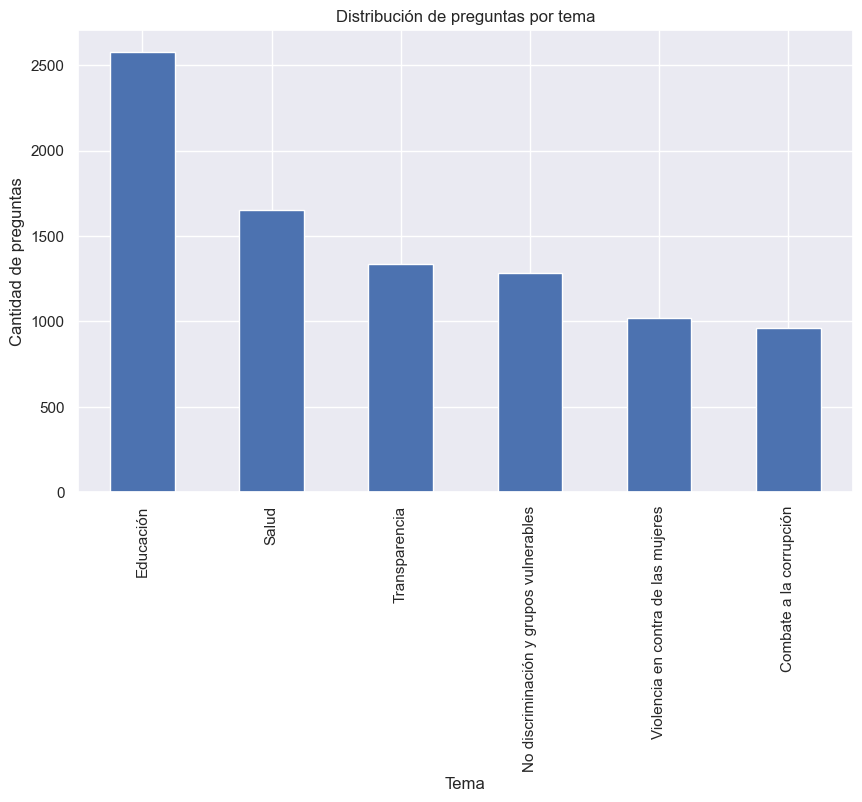
Al iniciar con la limpieza se vió que los encabezados de columnas tenían carácteres especiales que complicaban el uso de la base de datos con python. Estos se reemplazaron con encabezados más amigables. También se convirtió la fecha de texto a tipo de dato de fecha. También era el caso que cada línea podía contener hasta 6 preguntas, entonces el equipo las pasó a una sola línea por cada pregunta.

En la segunda libreta nos dimos cuenta de qué había muchas preguntas qué contenían palabras prohibidas, casi 1000 preguntas fueron eliminadas. Y las palabras qué tenían lenguaje de odio también tuvieron 336 preguntas eliminadas. Donde se eliminaron más preguntas fue en la detección si la pregunta iba acorde al tema o no, debido a que el algoritmo empleado demostró resultados relativamente aceptables. Sin embargo se pudo haber mejorado con una dataset más robusto y enfocado al estado actual del país. Con este método se logró eliminar más de 6 mil preguntas teniendo como resultado una base de datos de más de 16 mil preguntas

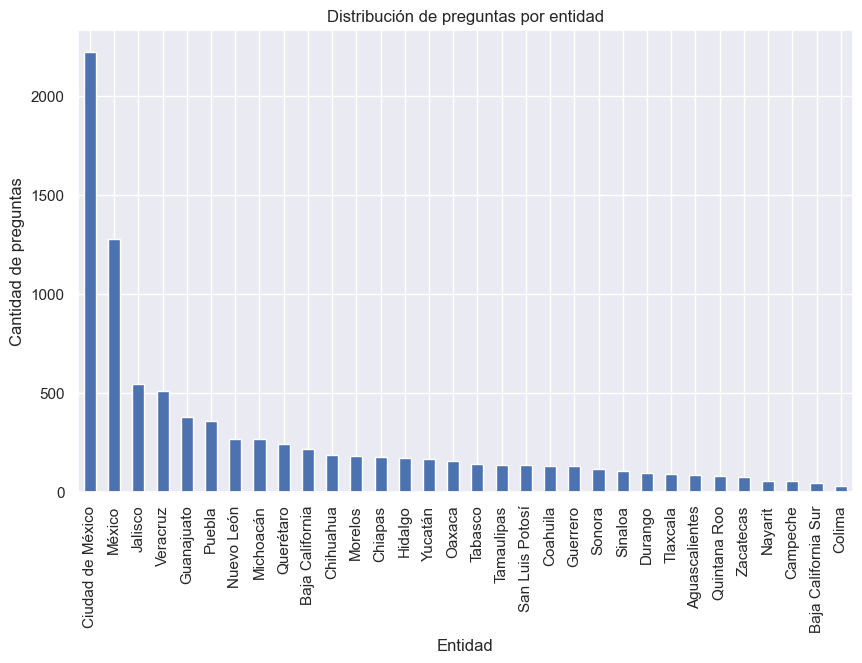
En la tercera libreta vino lo complejo, se eliminaron preguntas repetidas utilizando spacy, scikit-learn y el modelo de es\_core\_news\_md, normalizamos las preguntas con el corpus de stop words de nltk para posteriormente calcular la similaridad del coseno de cada pregunta y comparar cada pregunta contra todas las demás, si tenía una similitud mayor al .9 entonces se eliminaba la pregunta con menos antiguedad y además se guardaba en un archivo llamado 3\_preguntas\_repetidas.csv y las preguntas sin repetidas en 3-1\_base\_limpia.csv.

## Gráficas y explicación:

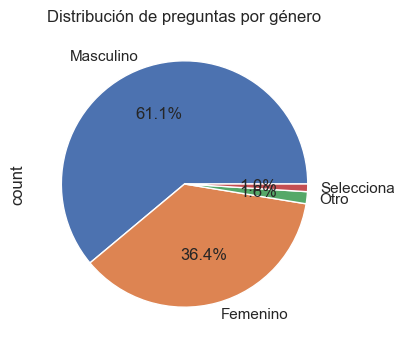
Los temas más preguntados ya después de la eliminación de preguntas qué no correspondían usando wikipedia.



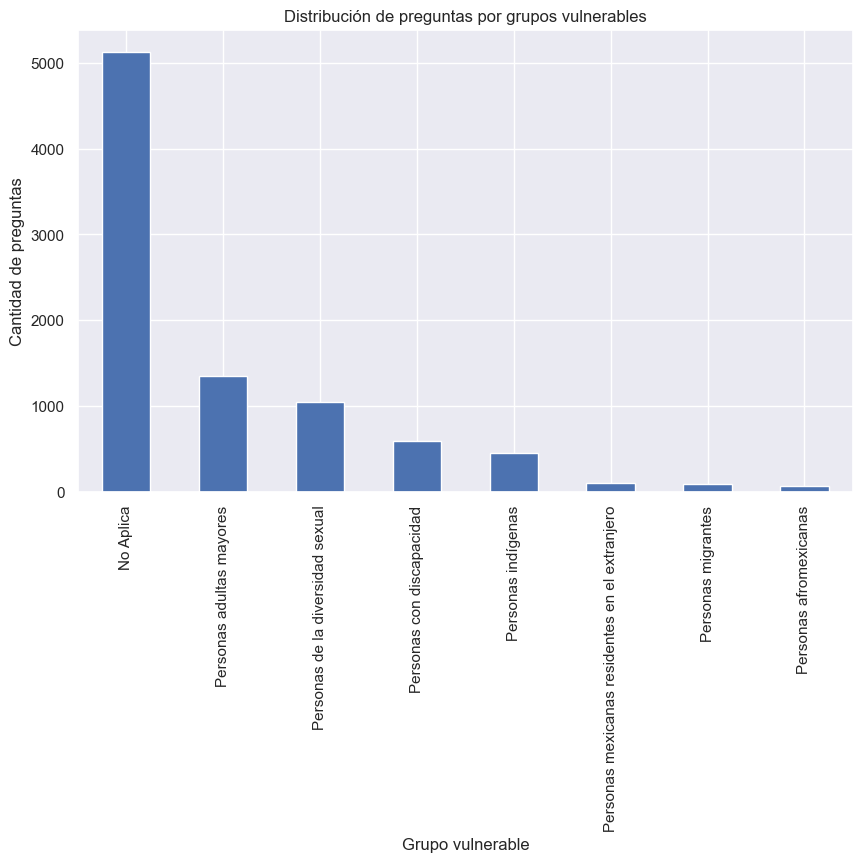
Las preguntas más repetidas por entidad.



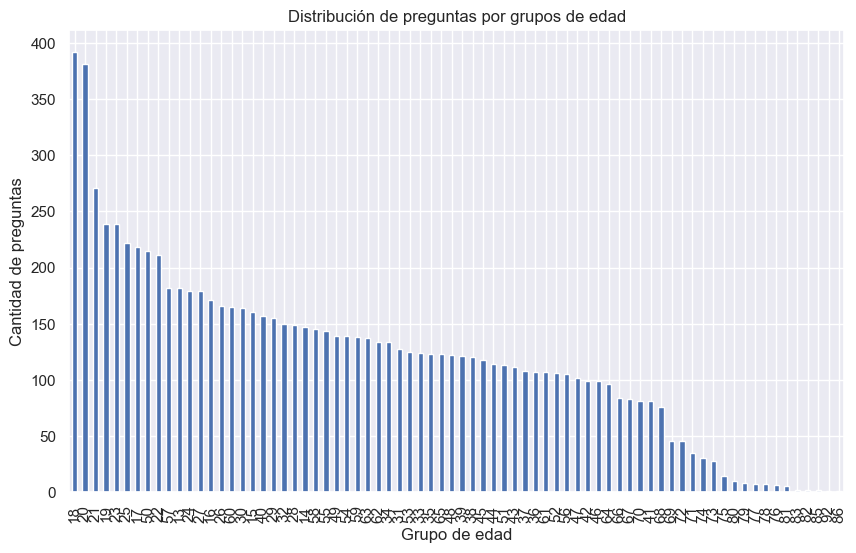
Distribución de preguntas por género:



Los grupos vulnerables y cuántas personas hicieron:



Preguntas por grupos de edad:



Nubes de palabras con palabras relacionadas a los temas más importantes:

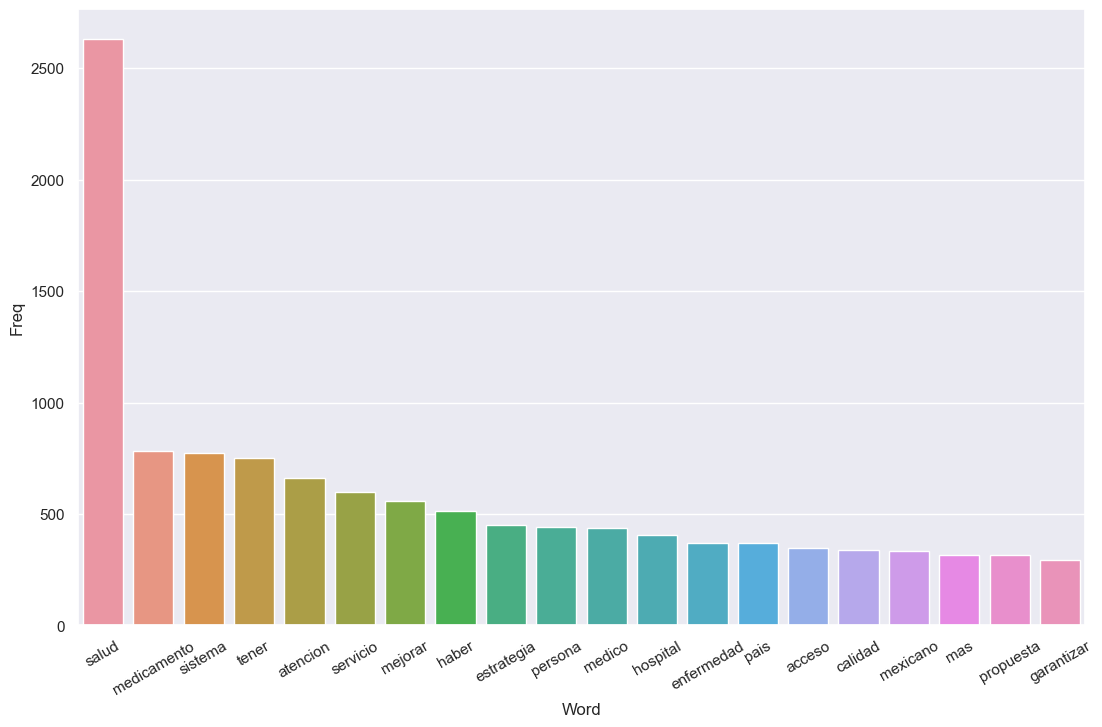
Nubes de palabras sin normalizar



Nubes de palabras normalizadas

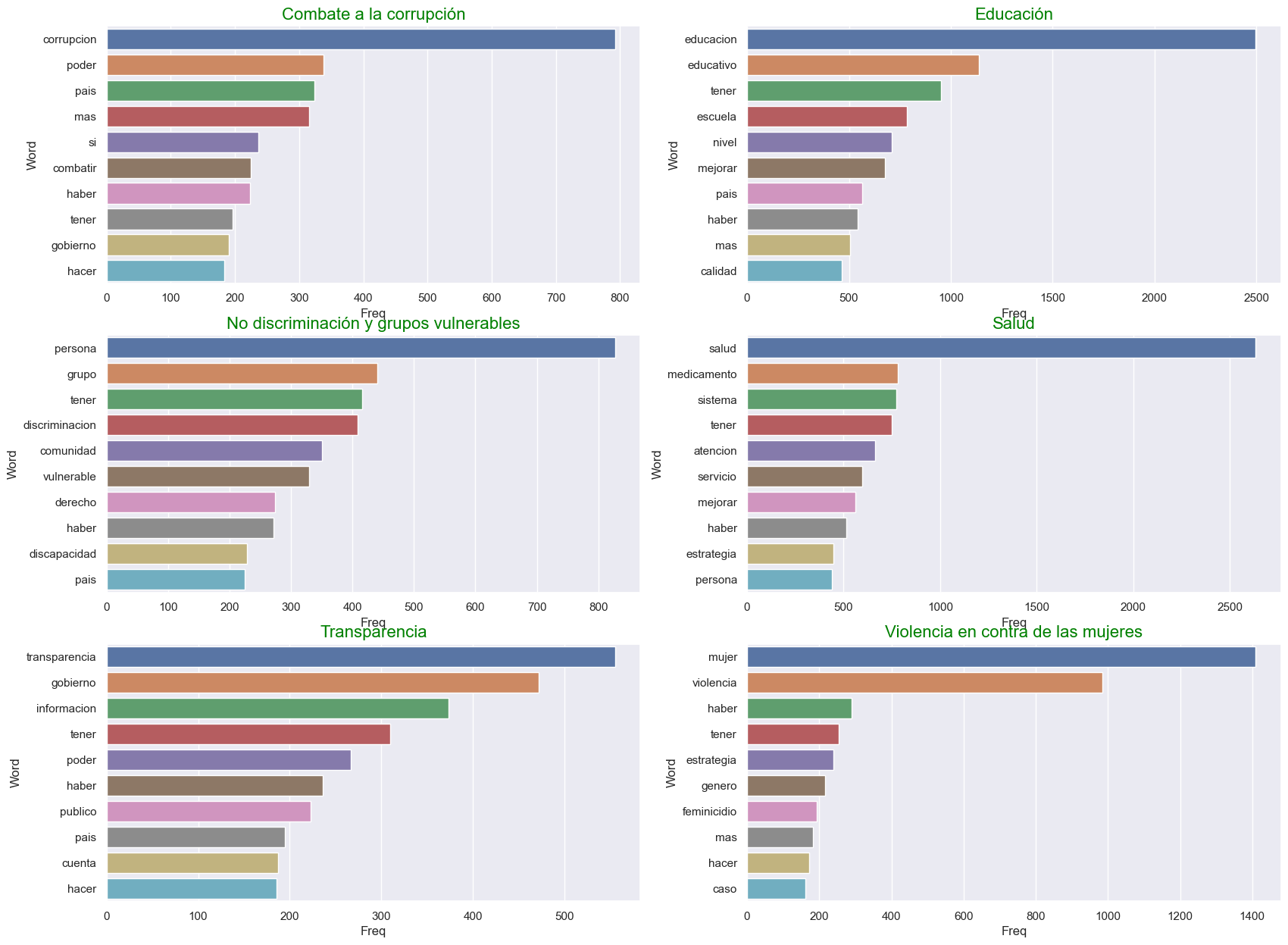
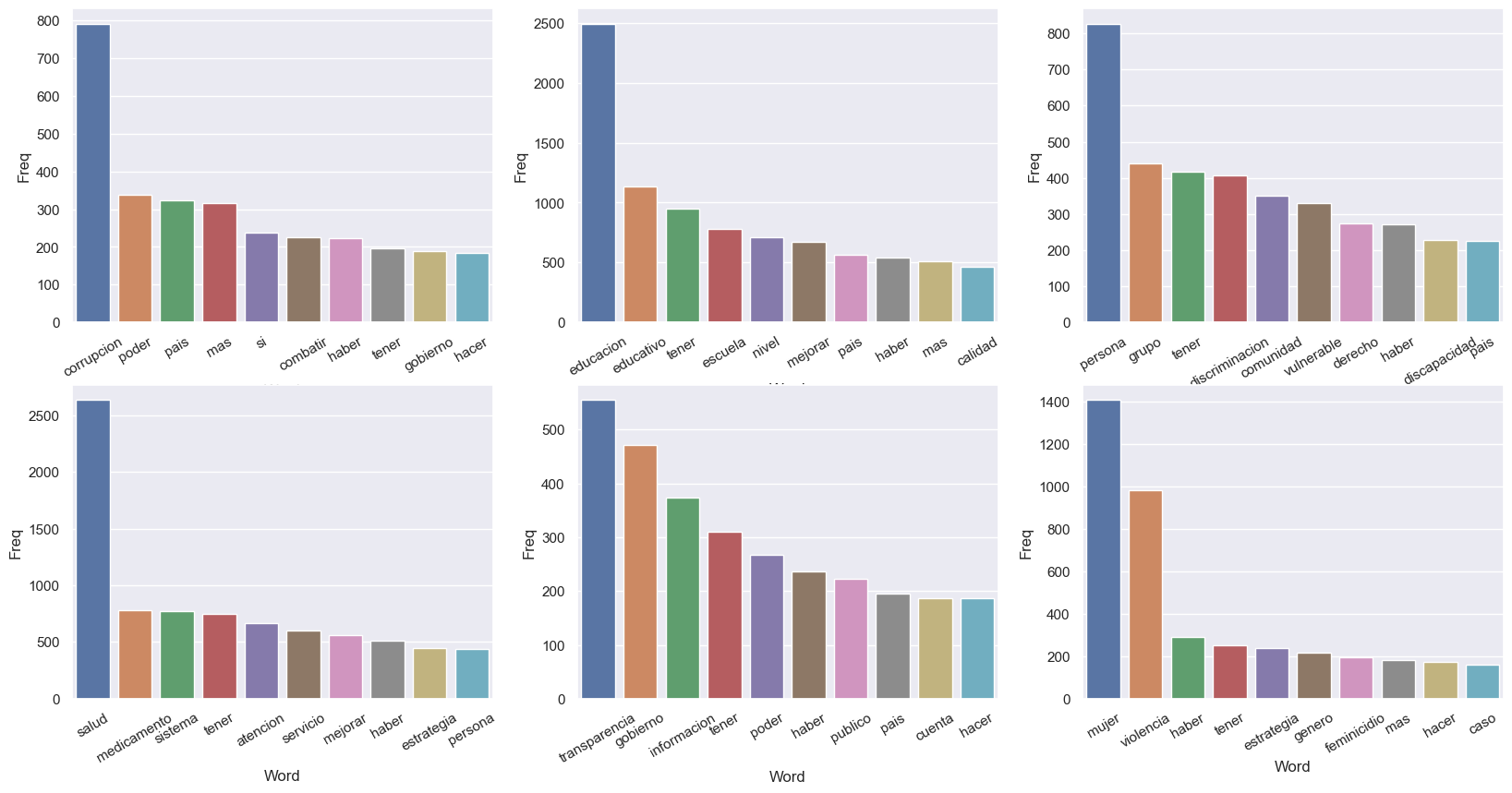


N-grama de palabras más repetidas en las preguntas de salud



Corrupción, educación, discriminación y grupos vulnerables, salúd, transparencia,

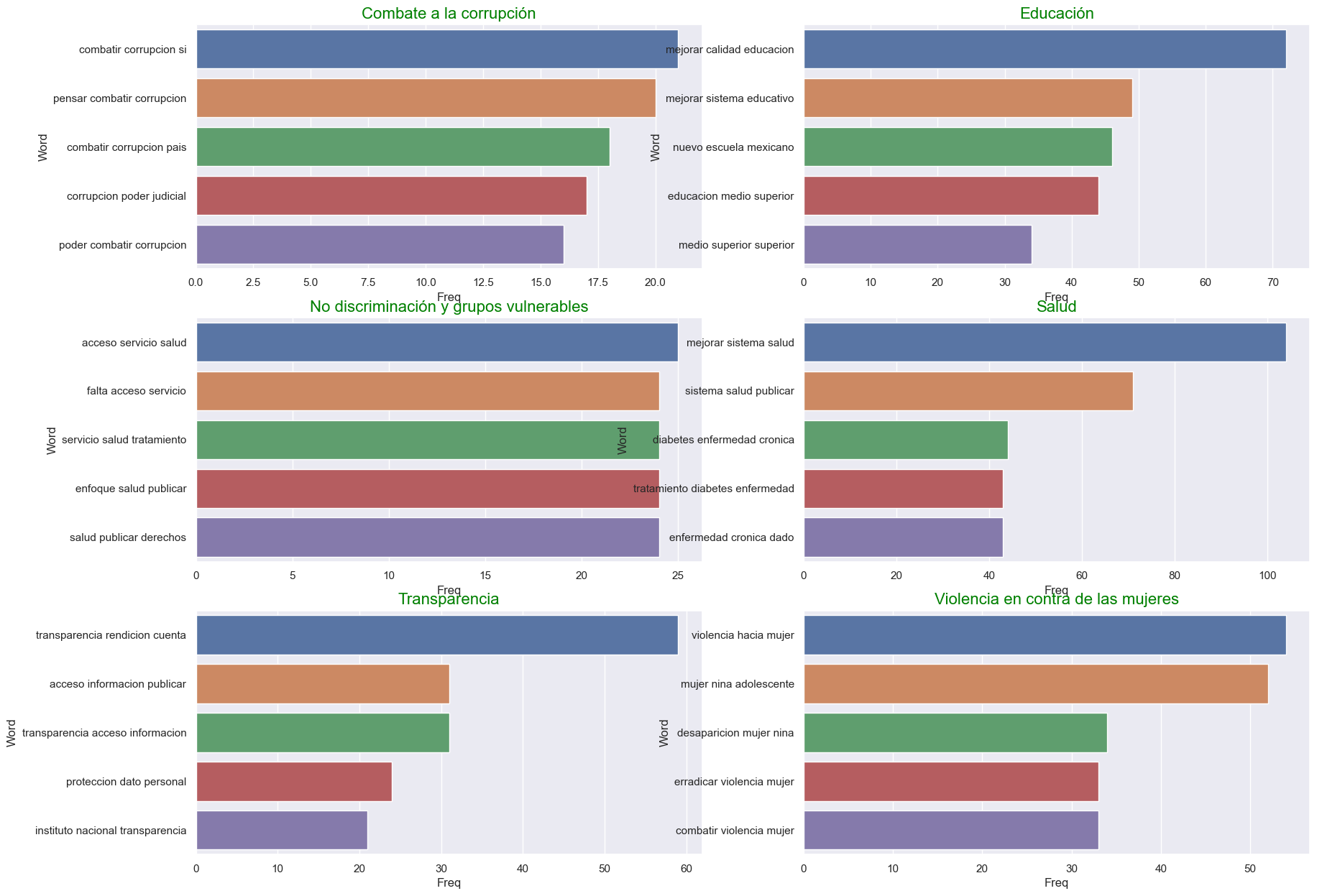
violencia contra las mujeres



Gráficas de bigramas sobre los temas de las preguntas



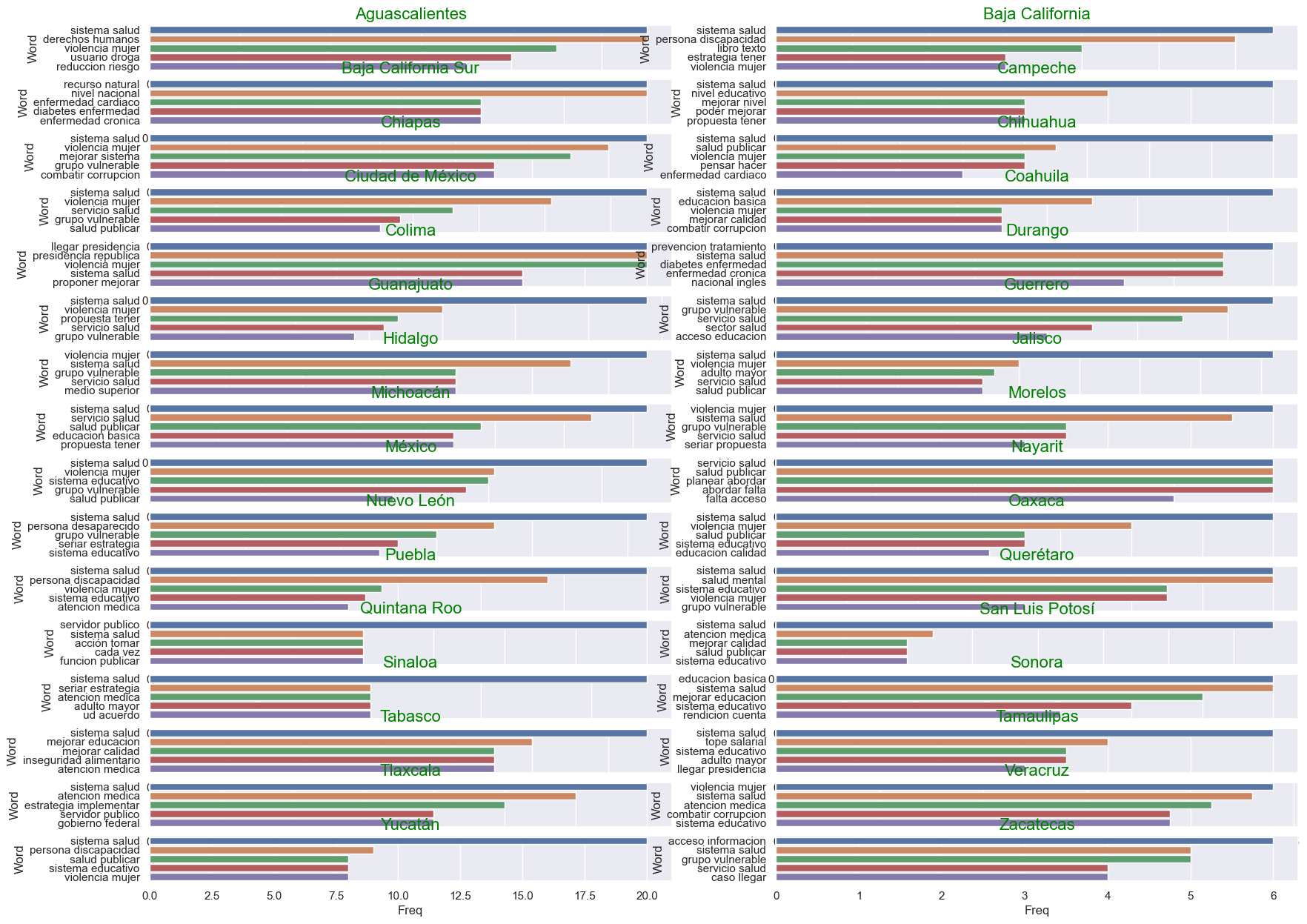
Gráfica con trigramas



Nube de palabras por estado con las palabras más repetidas en las preguntas



Bi-gramas por entidad

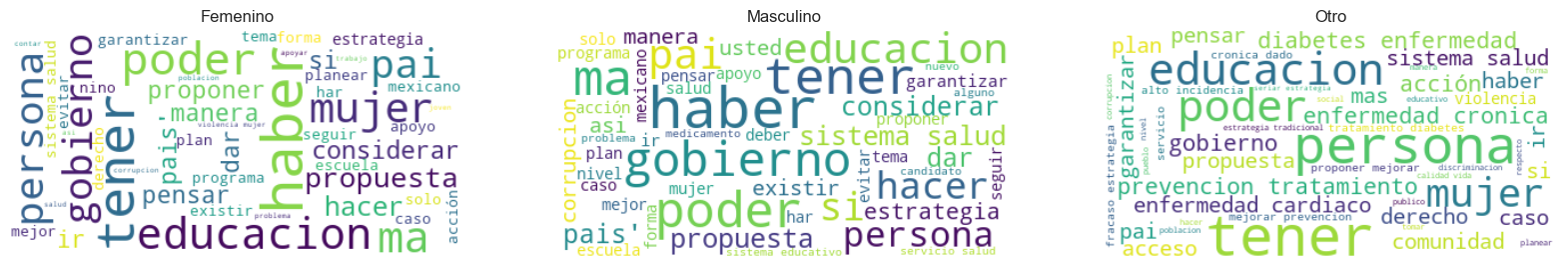


Tri-gramas por entidad

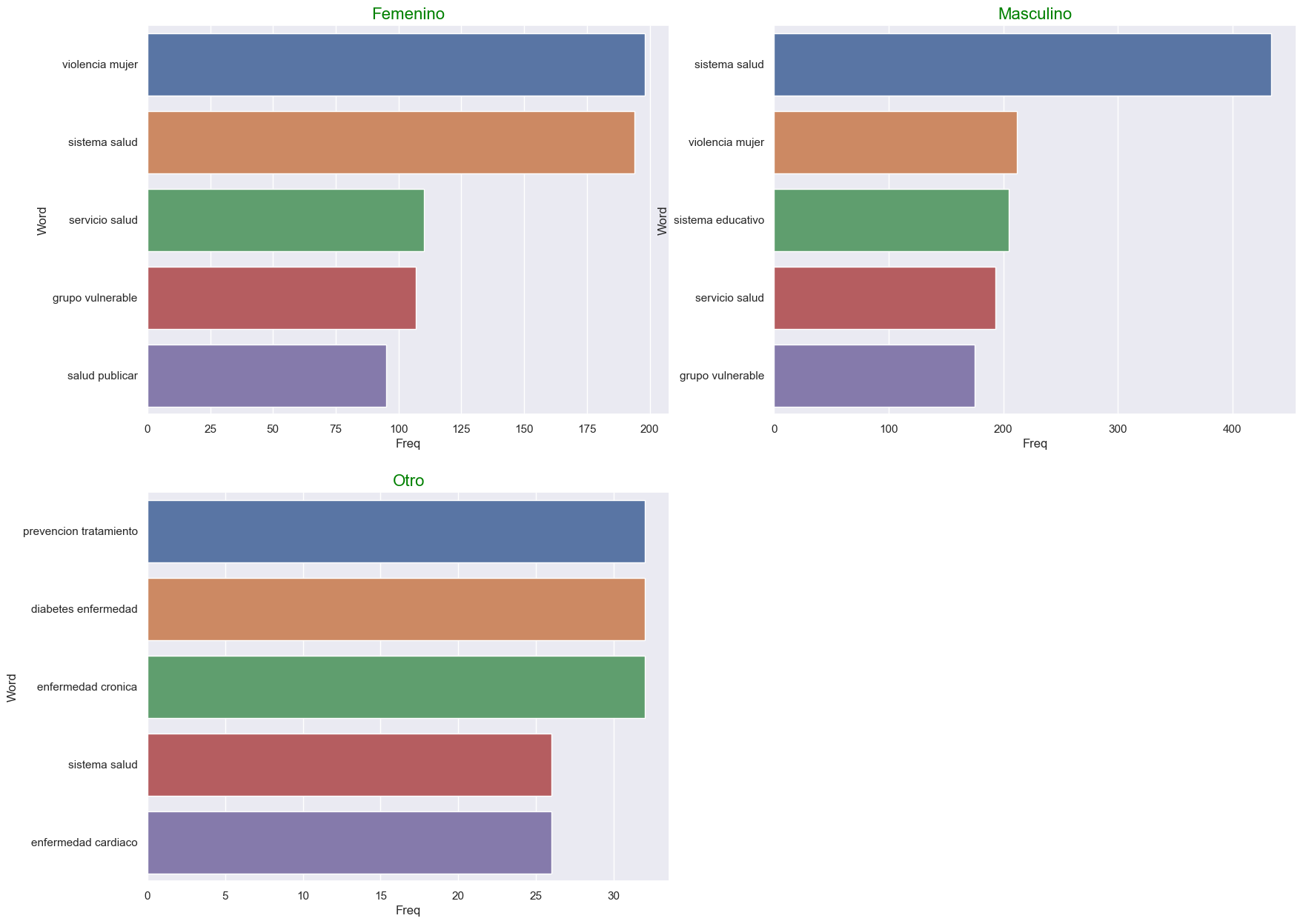


Por género

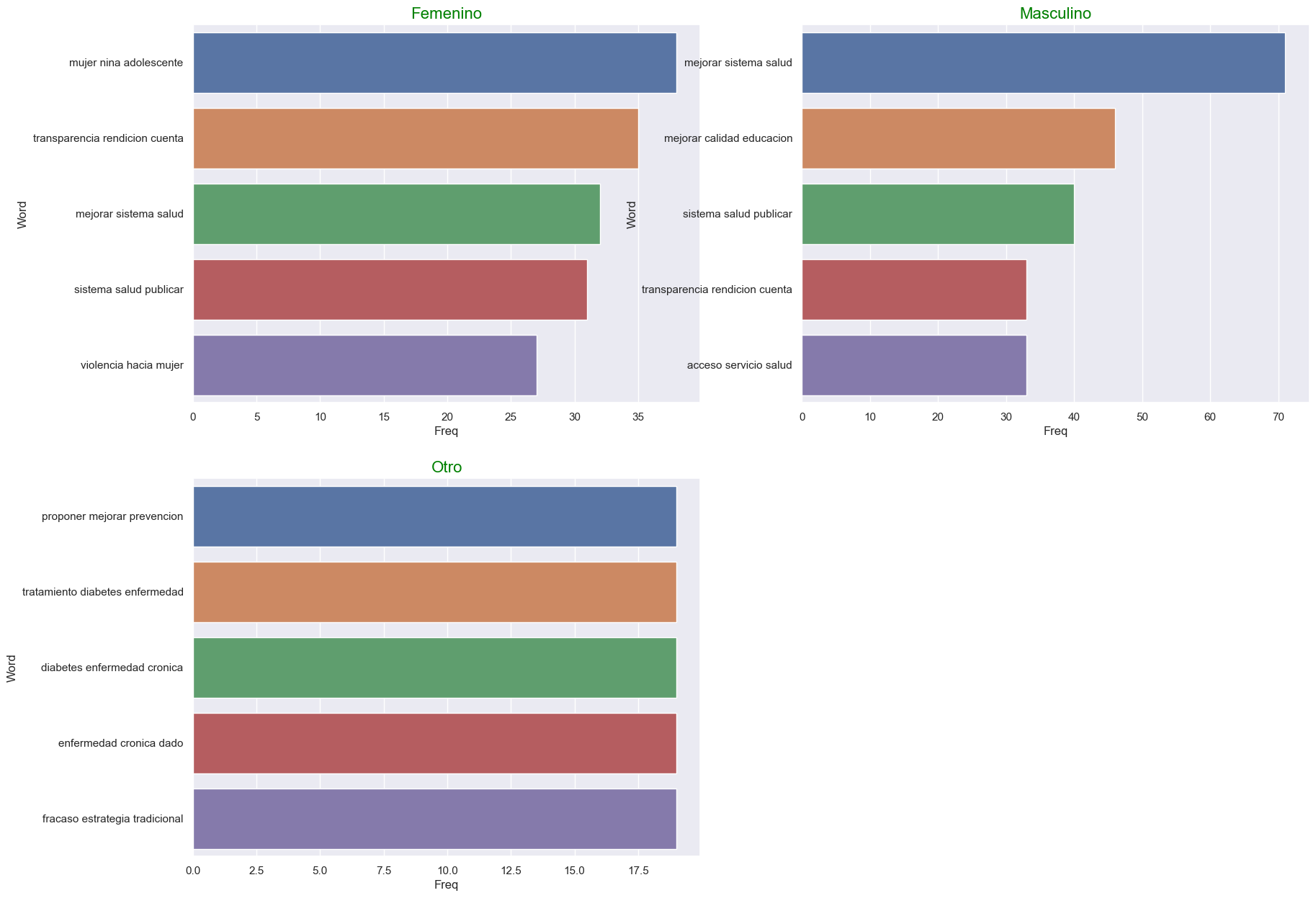
Nube de palabras



Bi-gramas



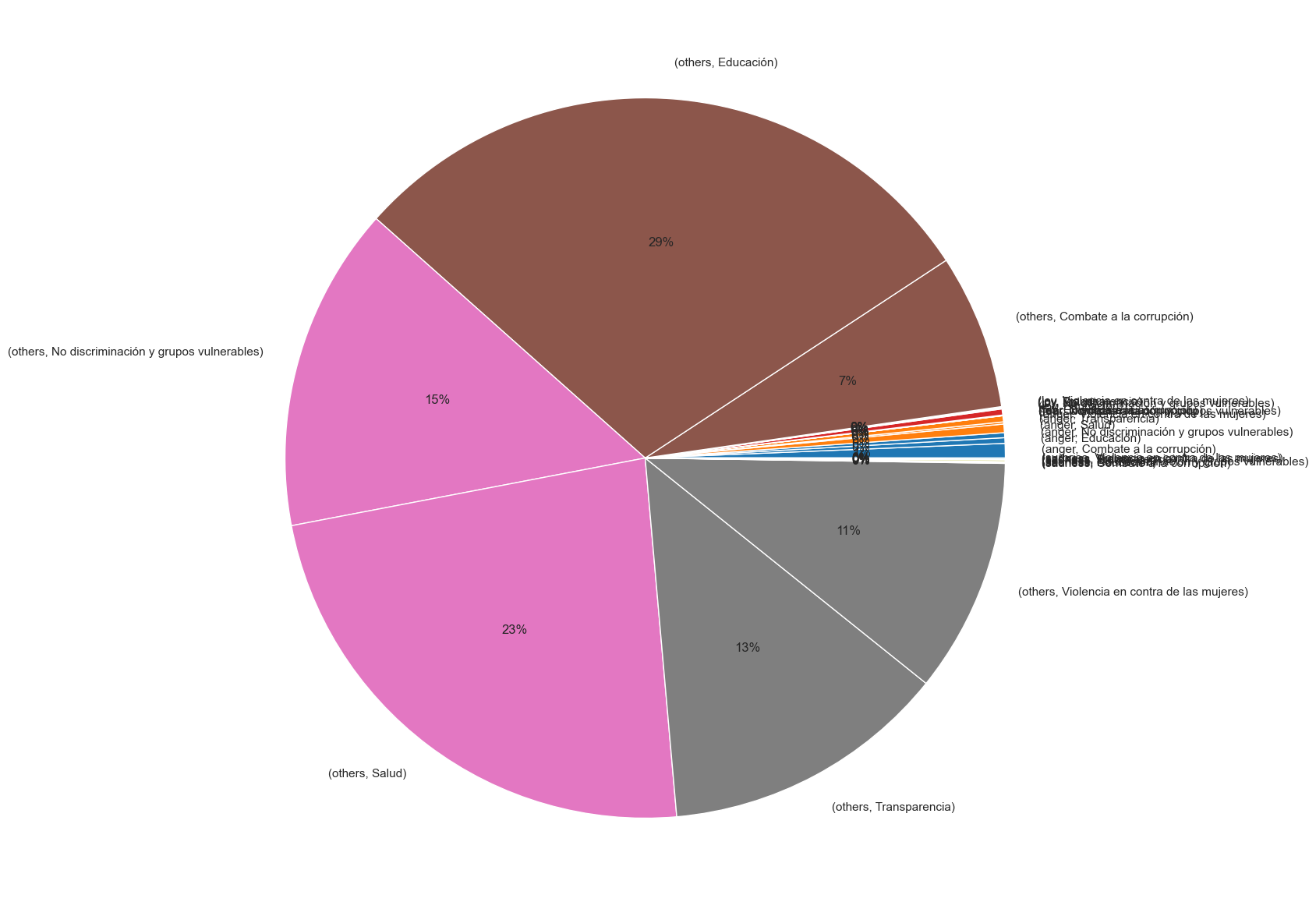
Tri-gramas



Análisis de sentimiento por temas



Análisis de emociones por temas



## Piezas de código que se usaron para generar esos resultados

Se usaron 4 notebooks donde se seccionó el trabajo hecho para analizar estas preguntas, en el primer notebook se utilizaron estas librerías:

* pandas (importado como pd)
* utils
* re

En el segundo notebook se utilizaron:

* pandas (importado como pd)
* spacy
* sklearn.metrics.pairwise (con la función cosine\_similarity)
* nltk
* re
* utils (con la función save\_df)
* pysentimiento (con la función create\_analyzer)
* math (con la función nan)
* unidecode
* numpy (importado como np)
* wikipedia
* collections (con la función defaultdict)
* itertools (con la función chain)

En el tercer notebook se utilizaron:

* pandas (importado como pd)
* spacy
* sklearn.metrics.pairwise (con la función cosine\_similarity)
* nltk
* re
* utils (con la función save\_df)

En el último notebook se utilizaron:

* **pandas**
* **sklearn.feature\_extraction.text (importado como CountVectorizer)**
* **transformers (con el nivel de registro establecido en ERROR)**
* **unidecode**
* **wordcloud**
* **matplotlib.pyplot (importado como plt)**
* **seaborn**
* **spacy**
* **nltk**
* **re**
* **pysentimiento (con la función create\_analyzer)**
* **transformers**

## Explicación general de los modelos utilizados

Para eliminar las preguntas que no estuvieran relacionadas con el tema seleccionado se hizo una ‘definición’ de lo que es un tema. Para definir que es un tema se hizo una búsqueda en wikipedia sobre artículos que creímos que eran abarcados por dicho tema. Por ejemplo, para el tema de salud buscamos las páginas de Wikipedia de “Salud” y “Medicina”; así para los otros 5 temas. Esta selección de artículos se hizo a mano. Después, de que se hiciera la búsqueda se seleccionaron las N palabras más utilizadas en cada tema y después este vector de palabras se utilizó para calcular la distancia de coseno entre esta definición de un tema y la pregunta en si. Si el resultado era mayor a un umbral se guardaba la pregunta y si esta no pasaba el umbral predefinido se descarta.

Para el análisis de odio el texto se utilizó crudo ya que la librería de *pysentimiento* ha demostrado, al menos en su README que puede procesar el texto crudo y con acentuación.

Para determinar si una pregunta era repetida de otra se usó un proceso de vectorización de usando una forma normalizada de cada pregunta, en puras minúsculas sin carácteres especiales. Si los vectores tenían una similitud basado en similitud coseno de 0.9 o más se contaba como una repetida y la pregunta más antigua se quedaba mientras que la más reciente quedaba descartada.

## Explicación general de las funciones definidas

Se definieron en su mayoría funciones que limpian texto, sea funciones que quitan acentos, signos de pregunta o que quitan stop words. Otra función notable es la que pasa de seis preguntas por línea en la base de datos original a una por renglón en la base de datos que se usó para trabajar.

## Conclusiones y aprendizajes

En este proyecto pudimos poner en práctica la mayoría de las herramientas aprendidas en este semestre y además pudimos implementar también las que obtuvimos en la materia de minería de texto.

La implementación del modelo de sentimientos fue exitosa y también la eliminación de preguntas repetidas lo que nos ayudó a hacer un buen análisis y pudiendo graficar una gran variedad de preguntas para entender y extraer la información más importante de este segmento de información.

## Bibliografía

Murzone, F. (2022, 30 marzo). Procesamiento de Lenguaje Natural. Preprocesado del texto 1: Tokenización | EscuelaDeInteligenciaArtificial. Medium. https://medium.com/escueladeinteligenciaartificial/procesamiento-de-texto-para-nlp-1-tokenizaci%C3%B3n-4d533f3f6c9b

Navarro, S. (2024, 12 abril). Similitud entre vectores o cosine similarity. KeepCoding Bootcamps. https://keepcoding.io/blog/similitud-entre-vectores-o-cosine-similarity/

Gerardo. (2023, 31 agosto). Stop Word: Qué es, Definición, Significado y Ejemplos ⋆ Sensacionweb.com. Sensacionweb.com. https://sensacionweb.com/diccionario/stop-word-que-es-definicion-significado-y-ejemplos/