INFORME DE PRÁCTICAS

MANUEL PALACIOS SÁNCHEZ UO282422

Universidad de Oviedo

JULIO 2023

Índice

APARTADOS OBLIGATORIOS

a. Datos del alumno

Nombre: Manuel Palacios Sánchez

DNI: 58437733T

Correo: uo282422@uniovi.es

Periodo de prácticas: Verano 2023

b. Datos de la empresa

Grupo de Investigación de Modelización de la Incertidumbre y la imprecisión en teoría de la decisión UNIMODE (Universidad de Oviedo)

Tutora académica: Noelia Rico Pachón

Tutora y representante de la empresa: S. Irene Díaz Rodríguez.

c. Descripción de la tarea y trabajos realizados

En cuanto a las actividades a desarrollar, se preveía la realización de un análisis de datos textuales obtenidos de Twitter sobre las publicaciones en esta red social de la comunidad de profesores de matemáticas. Con una recopilación, preprocesado y análisis de los datos relevantes, utilizando técnicas de minería de texto y Deep learning para la identificación de patrones, técnicas y comunidades de usuarios.

Entre las competencias a adquirir se esperaba la adquisición de competencias técnicas sólidas acerca de inteligencia artificial y análisis de datos, el desarrollo de habilidades en la comprensión de algoritmos de aprendizaje automático, la implementación de modelos de IA y la evaluación de resultados. También se pretendía adquirir competencias como la capacidad de seleccionar y aplicar técnicas apropiadas de minería de texto, y la capacidad de interpretar y comunicar de forma correcta los resultados obtenidos, teniendo un enfoque más práctico sobre la resolución de problemas reales en el ámbito de la informática.

Previo a las prácticas tuvieron lugar una serie de reuniones y correos para contar el propósito de estas y plantear los temas y trabajo a realizar.

Las prácticas se desarrollaron de forma telemática durante el mes de Julio de 2023, con la distribución de 6 horas diarias de Lunes a Viernes.

d. Competencias adquiridas y conclusiones

En lo personal, cuando se me presentaron estas prácticas me parecieron una buena oportunidad para conocer el mundo de la minería de datos y el Deep learning, que hoy en día tanta utilidad tiene. No era un ámbito que tuviese previsto trabajar o dedicarme en un principio, sobre todo debido al desconocimiento ya que, en el grado, al menos hasta el momento, no había tocado nada muy relacionado con el tema, por lo que decidí darle una oportunidad e intentar aprovechar este mes para aprender y desarrollar nuevas competencias.

La utilidad de este mundo, y la curiosidad por saber qué eran realmente estos conceptos de los que tanto se habla como la minería de datos, hicieron que me decantase por la realización de las prácticas y le diese una oportunidad.

Otro gran aliciente fue sin duda el ver y conocer más de cerca cómo es la investigación, ya que es algo de lo que no se habla directamente al pensar en prácticas en empresa y tenía curiosidad por saber como era el proceso de desarrollo.

Las practicas me han gustado bastante, tanto por los conocimientos adquiridos directamente de procesamiento de lenguaje natural y minería de datos como por las destrezas en investigación obtenidas. Respecto a esto último, me ha parecido curioso y desafiante la forma de obtener los conocimientos. En un principio, la idea que tenía acerca de las prácticas era que iban a ser más guiadas e iba e ir poco a poco, pero finalmente no ha sido así para nada ya que la mayoría de información fue obtenida de la lectura de publicaciones, capítulos de libros recomendados por Noelia o simplemente búsquedas en internet.

Esto al principio fue desafiante ya que al no ser lo que esperaba y no estar acostumbrado no me fue sencillo habituarme, pero creo que he aprendido mucho y desarrollado competencias en torno a la obtención y filtrado de información por mi cuenta. Además, me ha dado una visión muy positiva acerca de la investigación en informática y el planteamiento de una opción más de cara al futuro.

Hablando ya más de los conocimientos técnicos adquiridos, al principio me resultó algo bastante denso, seguramente una mezcla entre la novedad de la materia y la lectura en inglés, pero destacar lo interesante y útil que me parecieron sobre todo las técnicas de NLP. Estoy seguro de que en algún momento profundizaré mas sobre ello ya que creo que va a tener un papel fundamental en el mundo y me ha parecido muy útil aprender de ello, aunque sean conocimientos básicos.

En resumen y por concluir, me han parecido unas prácticas muy interesantes y las he aprovechado bastante bien, abriendo posibilidades extra para mi futuro como la investigación o el trabajo en IA, y se las recomendaría tanto a gente que parta de una base como a gente que, como yo, parta desde cero.

TRABAJO DETALLADO

a. Cronología de trabajo

Las prácticas comenzaron con la realización de diversos tutoriales en el portal de Hugging Face (https://huggingface.co/learn/nlp-course/chapter1/1) acerca de técnicas básicas sobre procesamiento del lenguaje natural, conociendo términos básicos de minería de datos como Transformers, modelos o tokenizadores entre muchos otros. Durante esta tarea he ido probando el código tanto en el Collab proporcionado en el tutorial como en el propio equipo para ir aprendiendo más como funcionan y en varios de los apartados sobre las tareas NLP he realizado notebooks resumen con funciones útiles del tutorial y breves explicaciones.

Tras esta primera semana de adaptación e introducción a la materia, he empezado a trabajar con la colección de tweets. Al principio me dediqué a la comprensión de la estructura de directorios y a identificar cómo se organizaba el contenido de cada archivo, analizando por ello la estructura de los Json de usuarios y de datos de cada consulta dada. De esta parte lo que más complicado fue, fue la organización en datasets y el limpiado de datos por cada tweet. También destacar que me costó al principio entender cómo tenía que tratar los tweets, si de forma individual por consulta o de forma global.

Posteriormente la siguiente tarea fue la de ir haciendo pequeños análisis sobre estos datasets de tweets, por lo que hice una serie de análisis y gráficos sobre las estadísticas de los tweets (número de likes, retweets, comentarios, citados).

Tras hacer esto, entré ya en materia en el procesado del texto de cada tweet y realicé un análisis de temas tratados y términos frecuentes, todo previa limpieza y filtrado de los textos.

Para terminar, lo último a lo que me dediqué fue al procesado de usuarios activos y realización de un grafo de relaciones. Esta última parte fue la que más me costó entender, sobre todo lo relativo al grafo ya que no estaba nada familiarizado y no me resultó nada fácil.

b. Entorno de trabajo

En un primer lugar, preparé la realización de las prácticas en una máquina virtual Ubuntu mediante VirtualBox, usando PyCharm para la ejecución de código. Tras unos días usando esto me di cuenta de que no me hacía falta para nada, de hecho, cuando tuve que ya empezar a trabajar con directorios tomé la decisión de trabajar directamente desde mi anfitrión y utilizar Visual Studio Code para ejecutar el Python. Esto supuso una mejora de rendimiento en comparación a trabajar desde la MV virtual, por lo que lo mantuve durante el resto de las prácticas.

Además, utlicé Google Collab para la realización y consulta de notebooks (sobre todo en la primera parte de Hugging Face). No conocía la herramienta y me ha parecido súper útil y cómoda para la toma de apuntes en la nube y la ejecución de código.

c. Primeros pasos con la colección

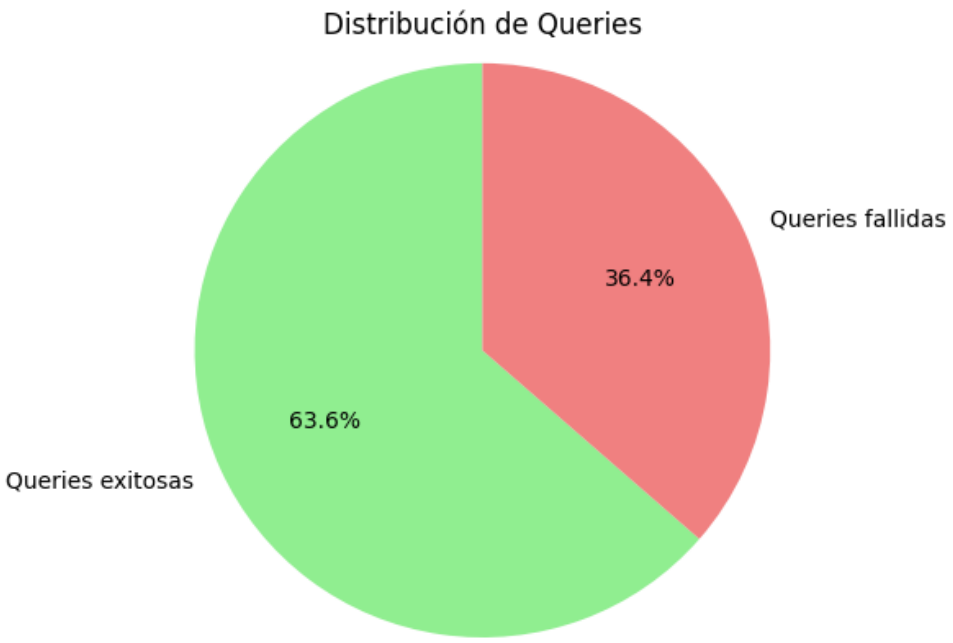
Destacar que se ha trabajado con la colección de tweets de 2019.

El primer paso, fue el de ir recopilando y juntando los datos en un dataset de la librería pandas. Para ello utilicé la siguiente función:

```python
#Este archivo recoge en un dataset todos los archivos de la colección de tweets

def obtener_archivos_recursivamente(ruta):
    datos_archivos = []

    elementos = os.listdir(ruta)

    for elemento in elementos:
        elemento_ruta = os.path.join(ruta, elemento)

        if os.path.isfile(elemento_ruta):
            datos_archivos.append({
                "Query": elemento_ruta.split("\\")[len(elemento_ruta.split("\\"))-2],
                "Nombre": elemento,
                "Ruta": elemento_ruta

            })

        elif os.path.isdir(elemento_ruta):
            archivos_subcarpeta = obtener_archivos_recursivamente(elemento_ruta)
            datos_archivos.extend(archivos_subcarpeta)

    return datos_archivos

ruta_raiz = r'C:\Users\Usuario\Desktop\Practicas\datos\tuits_2019\queries'

datos_archivos = obtener_archivos_recursivamente(ruta_raiz)

df_archivos = pd.DataFrame(datos_archivos, columns=["Query","Nombre", "Ruta"])  # Crear el DataFrame
```

Esta, organiza el dataset con la estructura [Query, Nombre, Ruta], siendo “Query” las palabras clave por las que se organizan los tweets, “Nombre” el nombre del archivo, y “Ruta” la dirección donde se encuentra el archivo. Este proceso lo hago debido a la estructura de directorios de la colección ya que por cada query hay una carpeta con varios archivos json (generalmente uno con datos de los tweets (data) y otro con datos de los usuarios (users).

Tras la creación de este dataframe estuve probando a manejar y manipular la estructura. Dado que no todas las querys tenían resultados, hice una pequeña función que me representaba el porcentaje de querys exitosas en forma de gráfico circular. Con la muestra actual:



Tras ver que había querys sin resultado, decidí pasar los datos a un nuevo dataframe con solamente los datos exitosos. La estructura del dataset es [“Query” (para no perder la consulta), “Data” (archivo de datos), “Users” (archivo de users)]

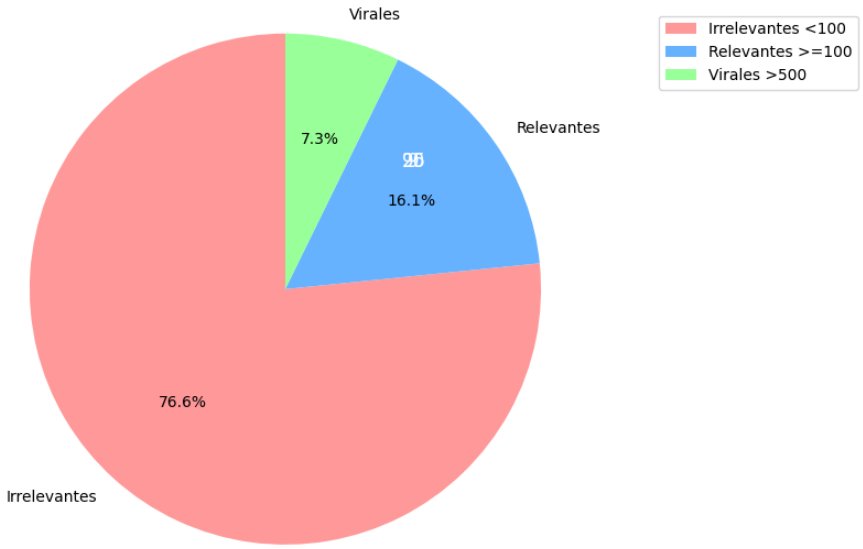
De esta forma, obtuve ya un dataframe con una estructura cómoda con la que poder trabajar más en profundidad.

d. Primeros análisis

Los primeros análisis que realicé fueron bastante simples. Saqué por ejemplo los valores máximos de likes de los tweets y realice una ponderación en función de el número de me gusta, retweets, comentarios y citados (todo bajo criterio personal) adjudicando una puntuación artificial para clasificarlos.

```
def calcularPondTweet(likes, rt, com, cit):
    score=0

    score=score+(5*cit)#cada citado cuenta x5
    score=score+(4*rt)#cada rt cuenta x4
    score=score+(3*likes)#cada like cuenta x3
    score=score+(2*com)#cada comentario cuenta x2

    return score
```

Partiendo de esta ponderación, y de los tweets más virales de cada query realicé una organización general a nivel de dataframe y saqué un gráfico, organizándolos en Irrelevantes, Relevantes y Virales. De esta forma el gráfico muestra que porcentaje de los tweets mejor ponderados de cada query, tiene más valor a nivel general.



e. Comienzo del procesado de textos

Tras los pequeños análisis mencionados anteriormente ya empiezo a tocar el análisis del contenido de los tweets en sí. Lo primero que hice fue sacar los textos del dataframe con los datos anteriores, además de otros atributos útiles del tweet como la fecha, el id del autor, el id del tweet o si es respuesta. Esta función filtra en función de los retweets, es decir, me he fijado en que los tweets retweeteados comienzan por “RT” por lo que mediante un simple parámetro.

```
def getTextos(datos, permitir_rt):
    textos_query = []
    for tweet in datos:
        tx = tweet["text"]
        f = tweet["created_at"]
        u = tweet["author_id"]
        i=tweet["id"]
        rt=""

        r_uid=""
        respuesta_a_user = tweet.get("in_reply_to_user_id", None)
        if respuesta_a_user is not None:
            r_uid=respuesta_a_user
        else: r_uid="Vacio"

        if permitir_rt == "no":
            if tx[:2] != "RT":
                textos_query.append((f, tx, u, r_uid,i))
        else:
            textos_query.append((f, tx, u, r_uid,i))
    return textos_query
```

Ahora que ya tengo esta función getTextos(), procedo a crear el dataframe con el que trabajaré directamente con los textos.

```
def crearDfTextos(permitir_rt):
    q_exitosas=get_df_exitosas_nombres(df_archivos) #DataFrame inicial con Query, Data, Users

    df_final = pd.DataFrame(columns=["Query","Id", "Fecha", "Mes","User", "RespuestaAUser", "Texto"])  # DataFrame final
    recuento_textos=[]
    for index, row in q_exitosas.iterrows():
        ruta = ruta_raiz + '\\' + row["Query"] + '\\' + row["Data"]
        with open(ruta, encoding='utf-8') as archivo:
            datos_q = json.load(archivo)


        tuplas_fecha_texto=getTextos(datos_q,permitir_rt)
        query=row["Query"]

        new_row={}
        for tup in tuplas_fecha_texto:
            if tup[1] not in recuento_textos:
                new_row = {"Query": query,"Id":tup[4], "Fecha": tup[0],"Mes":getMes(tup[0]),"User":tup[2],"RespuestaAUser":tup[3], "Texto": tup[1]}
                recuento_textos.append(tup[1])
                #print(len(df_final))
                df_final = pd.concat([df_final, pd.DataFrame([new_row])], ignore_index=True)


    return df_final
```

Siendo la estructura :

```
["Query","Id", "Fecha", "Mes","User", "RespuestaAUser", "Texto"]
```

De esta forma tengo organizado cada texto con la información que necesito, sin perder la query de origen.

f. Procesado final de textos

Una vez formateado el dataframe con las columnas con los datos que quiero, vamos a trabajar con los textos en sí. Al trabajar con textos, y sobre todo de fuentes coloquiales como pueden ser estos tweets, nos encontramos con gran cantidad de palabras no relevantes para analizar. Por esto, hemos de “limpiar” nuestros textos.

Primero lo que he hecho ha sido tokenizar usando una expresión regular, además he añadido que elimine las palabras que empiezan por ‘@’ ya que estas en el contexto de Twitter van a simbolizar menciones a otros usuarios. Una vez tenemos los textos ya separados en tokens’, vamos a eliminar otras palabras innecesarias. Eliminaré en primer lugar las palabras pertenecientes a la consulta ya que, por lógica, serán de las más repetidas. Después eliminaré palabras vacías del castellano añadiendo otras como ‘https’ ya que tras un primer análisis veo que son muy repetidas (en este caso concreto se refieren a los links).

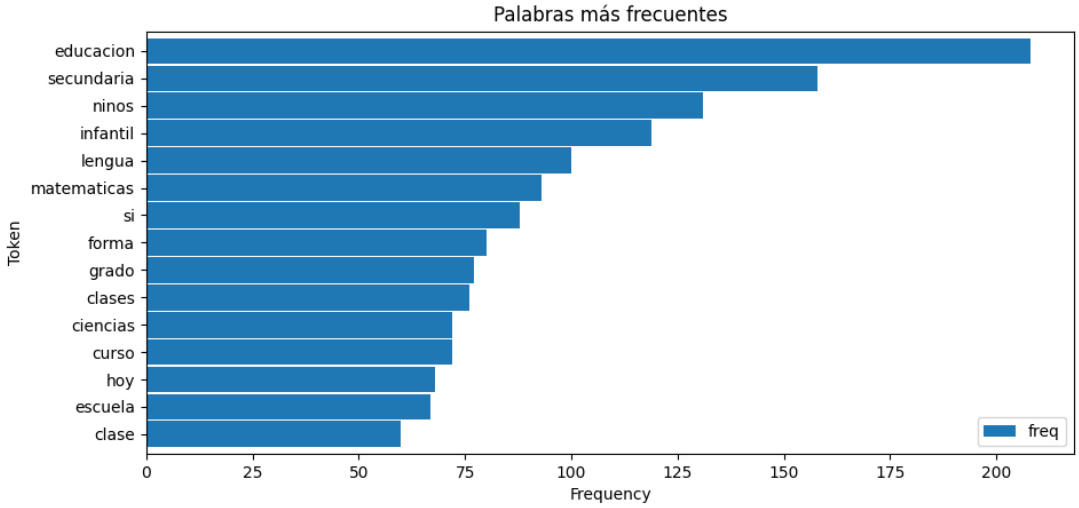
Otro aspecto a tratar y tener en cuanta debido a lo coloquial de Twitter, sería el pasar los caracteres todos a minúsculas y sin tildes, para así tener las palabras más sencillas de procesar.

Para aplicar todas estas funciones utilizaré un método prepare, a través de un array de tareas (a modo de pipeline)

g. Resultados del procesamiento de textos

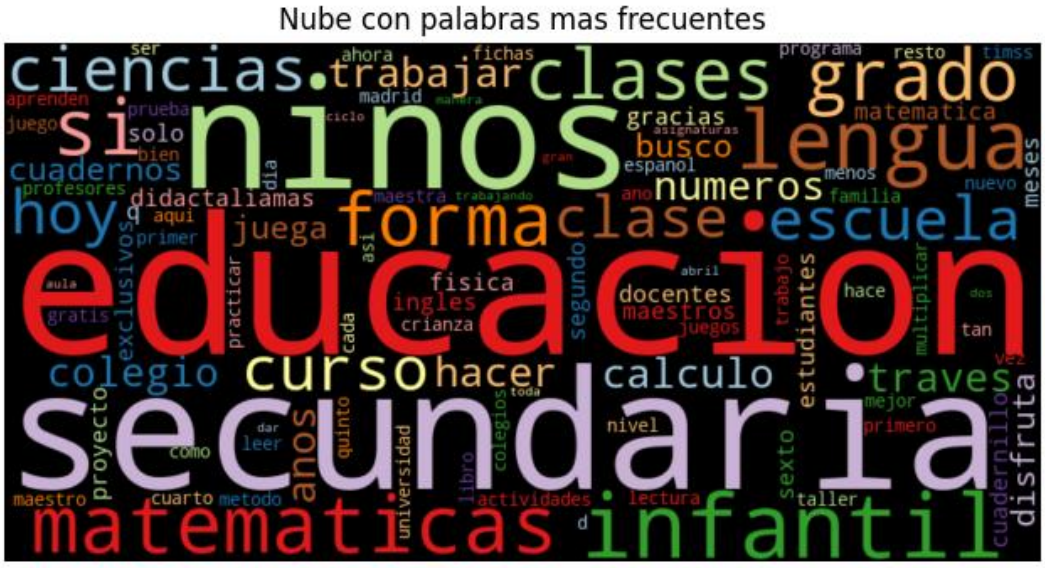
Una vez aplicadas las funciones de procesamiento de textos para dar lugar a una colección de tokens fiables podemos proceder a la generación de resultados y el cálculo de datos.

En primer lugar, genero un gráfico simple que mide la frecuencia normal de aparición de las distintas palabras (TF – term frecquency)



Podemos ver cómo claramente hay términos que se repiten gran cantidad de veces más que otros, incluso dentro de este top de palabras.

Como segundo gráfico, genero una nube de palabras, también basándome en la aparición de los tokens, por lo que deberíamos obtener resultados similares en un formato distinto.



Efectivamente, vemos en más grande los términos más aparecidos. Esta forma de representación es bastante más visual que un gráfico simple, por lo que trabajaré con esta misma para el ajuste de las frecuencias. Si nos paramos a pensarlo, las veces que aparece una

palabra no es del todo representativo, por lo que aplicaremos las formulas matemáticas para calcular las frecuencias teniendo en cuenta la IDF (inverse term frecquency) y la TF, que nos sirven usándolas en conjunto como un baremo para determinar las palabras más relevantes.

Para mostrar este "refinamiento", comparé las palabras más relevantes de los meses enero y octubre usando por un lado la TF y por otro la TF-IDF.

Enero - TF

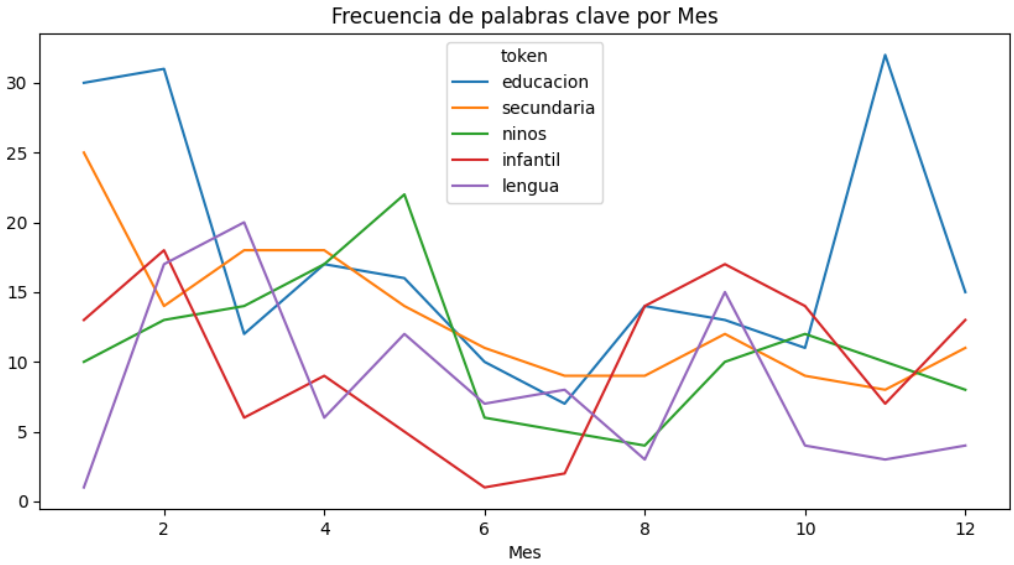
Octubre - TF

Enero - TF-IDF

Octubre - TF-IDF

Los resultados a pesar de no ser muy distintos, difieren mucho, dando así una nueva alternativa sobre qué tener en cuenta para ver la relevancia de las palabras. Destacar que refinando más las palabras vacías podemos obtener mejores resultados (En caso de TF-IDF de octubre por ejemplo añadiendo a la lista de palabras a eliminar 'h' o 'gran').

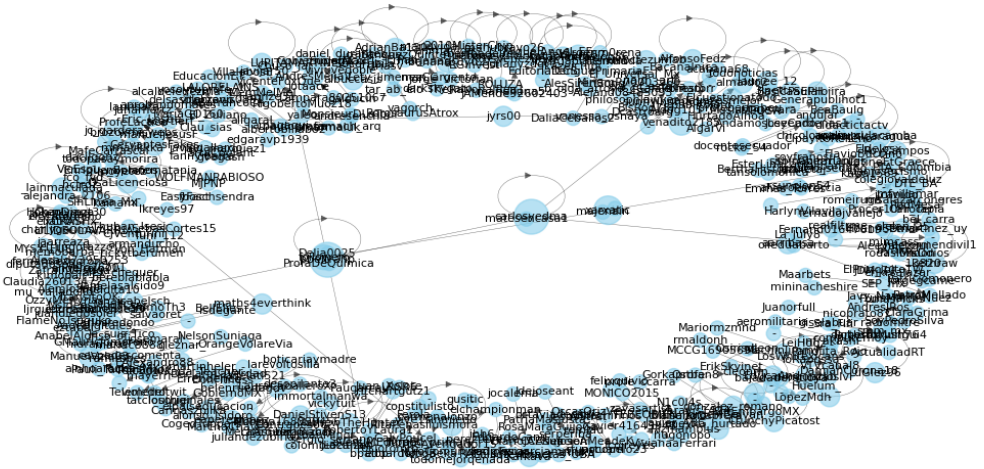
Por último para este análisis de tokens generé un gráfico que muestra la evolución de la frecuencia de las palabras con el tiempo mes a mes. Utilizando un gráfico de líneas donde en el eje de las X tenemos la evolución temporal frente al eje de la Y con la frecuencia.



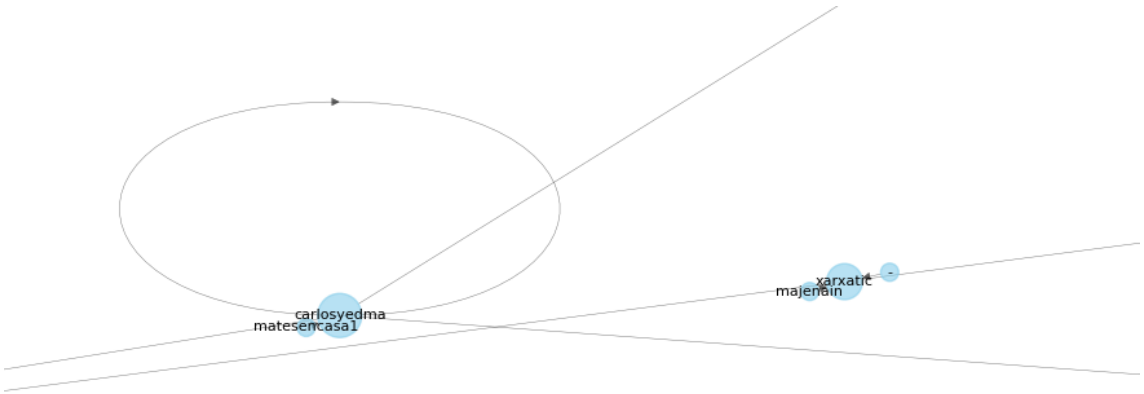
h. Procesado por usuarios

Para el procesado de usuarios, he creado un método que me devuelve un simple dataframe con la estructura [“Id”, “Nombre”], usando para ello el archivo json de users.

Generé grafo dirigido en función de las respuestas de los usuarios y las interacciones entre los distintos perfiles.



Haciendo zoom por ejemplo a los dos nodos centrales más a la derecha:

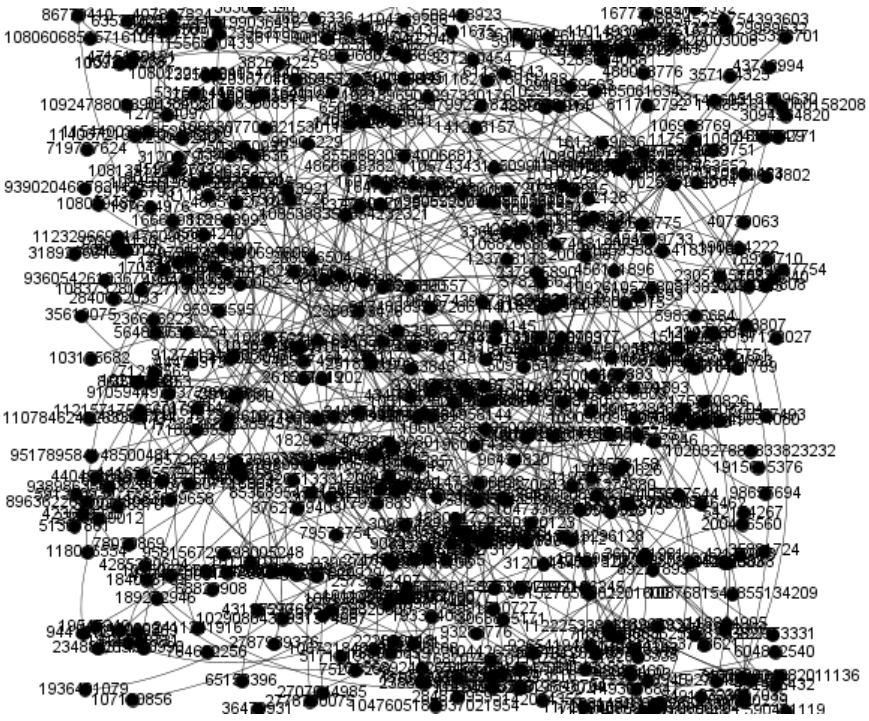


En un principio, y siguiendo uno de los posts recomendados, lo que hice fue guardar el grafo en un archivo .graphml y posteriormente analizarlo con el programa Gephi, pero no me aclaré nada bien y decidí hacerlo directamente usando la función:

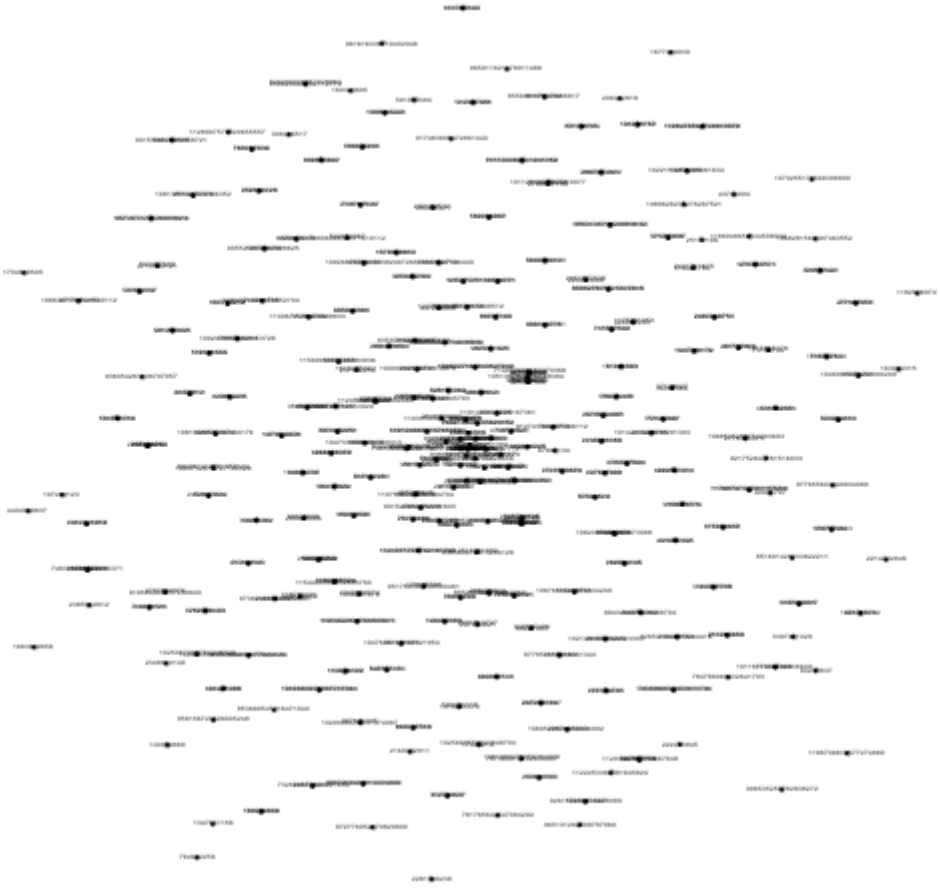
```python
def crearGrafo2(df):
    G = nx.from_pandas_edgelist(df, source='User', target='RespuestaAUser', create_using=nx.DiGraph())
    sizes = [x[1] * 100 for x in G.degree()]

    # Dibujar el grafo
    pos = nx.spring_layout(G, seed=42)

    nx.draw_networkx(G, pos, node_size=sizes, with_labels=False, alpha=0.6, width=0.3, node_color='skyblue')

    # Agregar etiquetas a los nodos
    labels = {node: traducirIdUser(node) for node in G.nodes()}

    nx.draw_networkx_labels(G, pos, labels, font_size=8, font_color='black', verticalalignment='center')

    plt.axis('off')
    plt.show()
```

Esta función coloca a cada usuario como arista y le coloca una label que muestra el nombre del usuario en vez de su id, para tener así un poco más claro los nombres.

El grafo generado permite visualizar las relaciones entre usuarios mediante sus interacciones, aunque no tan claramente como se pretendía en un principio. Esta es la parte que más me ha costado, ya que el trabajar con tantos usuarios diferentes sumado a no apañarme muy bien con lo que es la creación de este tipo de grafos en sí, ha hecho que me demore más de la cuenta para sacar incluso este resultado. Aquí muestro algunas capturas de lo que fue el proceso que intenté seguir en primer lugar usando la herramienta Gephi, pero que decidí abandonar y continuar con la alternativa anterior.



Aplicando un Layout Atlas iba mejorando algo pero nada que me dejase conforme.



ANEXO 1. REFERENCIAS BIBLIOGRÁFICAS

Como referencias bibliográficas proporcionadas he utilizado los siguientes libros de texto y las siguientes webs:

- https://huggingface.co/learn/nlp-course/ : Curso tutorial de Hugging Face como introducción al NLP
- Python for data Analysis (O Rilley)
- Blueprints for Text Analytics Using Python
- https://jrashford.com/2022/10/14/building-a-social-network-from-retweets-in-python-a-simple-guide/
- https://medium.com/social-media-theories-ethics-and-analytics/analysis-of-twitter-social-network-d5023e1a1aa
- https://campus.datacamp.com/courses/analyzing-social-media-data-in-python/twitter-networks?ex=4

Además de otras búsquedas a webs variadas por más información.

ANEXO 2. RECURSOS PROPIOS

Como recursos propios adjunto enlaces a varias notbook de Collab y al repositorio de GitHub con el archivo con el que he trabajado. Acerca de este último destacar que la ruta a los directorios no funciona ya que es donde estaban en mi caso.

Notebooks de realización de tutoriales Hugging face.

- https://colab.research.google.com/drive/1Z3yXIyQ7CVzHxGja1f2olXvN6UiiK6cy?usp=sharing
- https://colab.research.google.com/drive/1bL1ogvNUMHrQdp1fJQbRzdlH6k6eAqEp?usp=sharing
- https://colab.research.google.com/drive/1O4aoohtoJj5kq6jqQSCFWqstP-bRZObW?usp=sharing

Repo GitHub.

- https://github.com/uo282422/PracticasManuelPalacios