TRABAJO FIN DE GRADO

INGENIERÍA INFORMÁTICA

Vigilancia Tecnológica y Minería de Opiniones en RRSS

https://github.com/mikykeane/TFG/

Autor

Miguel Keane Cañizares

Director

Antonio Gabriel López Herrera

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—

Granada, Septiembre de 2019

Vigilancia Tecnológica y Minería de Opiniones en RRSS

Miguel Keane Cañizares

Palabras clave: RRSS, Twitter, Netflix, HBO, Streaming, Meaningclud, Wordcloud, Tweepy, MongoDB, Pymongo, Python

Resumen

Hoy en día las redes sociales son la mayor fuente de información en existencia, no en calidad, pero en cantidad. Eso significa que la cantidad de información es altísima, lo cual no significa que sea de utilidad, puesto que debido a su volumen es imposible de analizar para un individuo. Por ello surgen avances cómo el análisis de sentimientos, para intentar extraer información subliminal de textos de forma automatizada, es decir, sin intervención humana. Esto es parte de lo que llamamos minería de opiniones, analizar la información proporcionada por los usuarios y descifrar el significado latente de sus palabras idealmente como podría hacer una persona. Esto hace que la gran cantidad de información pueda ser también de calidad.

Este proyecto se centrará en la obtención y el análisis de esta información que hay disponible en las redes sociales y convertir un grueso de información bruta en datos útiles que sean analizables y puedan proporcionar conclusiones prácticas para individuos o empresas.

Technological Surveillane and Opinion Mining

Miguel Keane Cañizares

Keywords: Social Network, Twitter, Tweepy, Streaming, MeaningCloud, WordCloud, MongoDB, Pymongo, Python, Netflix, HBO

Abstract

Nowadays social networks have become the main source of data in the world, but it's not quality information, which means that the amount of data is enourmous but that doesn't mean it's useful information. Because of its high volume it's impossible for an indivuald or even a group of individuals to analize it all. That's where Sentiment Analysis steps right in, to extract subyacent data from texts in an automated procedure without human inteference. This is what we call Opinion Mining, to analyze the information given to us by the users and decipher it's meaning as a person could do. This would make the data into quality data.

The aim of this project is to obtain and analyze the data that's available in social network and turn a huge pile of raw data into something useful that can be analyzed and provide critical or at least practical information to indivuals or companys.

Yo, **Miguel Keane Cañizares**, alumno de la titulación Ingeniería Informática de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 76656535L, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Miguel Keane Cañizares

Granada a 5 de Septiembre de 2019 .

D. **Antonio Gabriel López Herrera)**, Profesor del Área de XXXX del Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado ***Vigilancia tecnológica y minería de opiniones***, ha sido realizado bajo su supervisión por **Miguel Keane Cañizares**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 5 de Septiembre de 2019 .

El director:

Antonio Gabriel López Herrera

Agradecimientos

He de agradecerle el presente a mi familia por su inestimable apoyo, a mis profesores por su profesionalidad y dedicación, a mis amigos, sin los cuales este proyecto hubiese estado terminado mucho antes y sobretodo a StackOverflow, sin el cual nada de esto hubiese sido posible.

Índice general

Índice de figuras

Capítulo 1

Introducción

> "Ya no estamos en la era de la información. Estamos en la era de la gestión de la información"
>
> Chris Hardwick, actor

La llamada Big Data, es reina indiscutible del futuro del análisis de información. Antiguamente, el problema solía ser la falta de información disponible, pero hoy en día, el problema es que disponemos de más información de la que nadie sería capaz de procesar. Por ello, debemos automatizar dicho procesamiento, crear programas que extraigan y analicen la información a nuestro alcance para así obtener una información estadística que nos sea de utilidad, ya sea para análisis estadísticos, marketing o satisfacción del cliente. Con la información correcta se pueden tomar las decisiones correctas.

A este proceso de obtención y análisis de información, lo llamaremos Minería de Opiniones. Cuya finalidad será conocer que opina un gran número de personas sobre el tema deseado mediante lo que comparten en las redes. Y esta será la finalidad de este trabajo de fin de grado, orientarlo al análisis de redes sociales, usar y cuando sea necesario, crear, herramientas que permitan acceder a la información, permitan analizarla y enfocarla hacia una finalidad práctica.

1.1. Motivación

Debido al auge de las redes sociales en los últimos años y los grandes cambios sociales que estas han conllevado, analizar la Big Data que nos llega de estas plataformas se ha convertido en uno de los grandes imprescindibles para todas las grandes y medianas empresas. Por ello, siendo un tema de interés y actualidad he querido trabajar en este proyecto, el cual estará centrado en obtener información de las redes y analizarla de forma que se obtenga

información que pueda serle de utilidad a una empresa. Además, dentro de la aplicación, también ha sido una fuerte motivación el hecho de poder hacer este proyecto en python, puesto que deseaba mejorar aptitudes en este lenguaje de programación.

1.2. Definición del problema

La información que nos llega de las RRSS es abrumadora, la finalidad de este proyecto será su obtención y posterior análisis.

1.3. Redes Sociales

Una red social es una estructura social compuesta por un conjunto de usuarios que están relacionados de acuerdo a un criterio. Y es un concepto que se empezó a utilizar a comienzos del s.XX, donde Georg Simmel lo utilizó para referirse a los vínculos que unían comunidades, defendiendo que el grupo no crea los vínculos, sino que los vínculos crean el grupo. En la actualidad, cuando nos referimos a una red social solemos referirnos a una u otra de las múltiples plataformas que existen en Internet. Y aunque se podría decir correctamente que Instagram, por ejemplo, es una red social, lo cierto es que dentro de Instagram se crean múltiples redes sociales, grupos de gente con intereses comunes como puede ser un hashtag, donde la gente se junta para participar en discusiones sobre un ámbito u otro. Dentro de las plataformas vigentes, algunas de las más relevantes actualmente serían:

Instagram

Instagram (2010) es la red social de moda entre los jóvenes, sus comunidades giran en torno al hashtag, los cuales son palabras precedidas por una almohadilla (#), con las cuales los usuarios pueden encontrar un sinfín de publicaciones sobre el tópico concreto de la almohadilla. Siendo sitio preferido por los llamados influencers, los cuales son personas debido a su alto perfil en las redes y elevado número de seguidores, poseen una cierta influencia sobre la red y pudiendo llegar incluso a generar ingresos gracias a la publicidad. Esta red casi fue la elegida para ser analizada en este proyecto, pero debido a que la mayor parte del contenido es en forma audiovisual o fotográfico, suponía una complicación añadida a la hora de minar opiniones.

Facebook

Facebook (2004) aunque no el origen (la primera red social fue SixDegrees, 2001), si es el causante de la masificación de las redes sociales en Internet. Es la red social con más usuarios en todo el mundo y dueña de las otras más cotizadas, como Instagram y WhatsApp. Esta red social fue otra de las

grandes candidatas a ser objeto de la minería de opiniones de este proyecto, pero debido a su carácter privado, donde la gran mayoría de la gente tiene el perfil cerrado para que solos sus amigos puedan acceder a su contenido, suponía una dificultad insalvable a la hora de obtener un tráfico de información aceptable para el estudio.

Twitter

Twitter (2006) fue y sigue siendo una de las redes más relevantes en la actualidad, y la que será objeto de estudio en este proyecto, debido a que es utilizada por gente de todo el mundo para la discusión de temas de actualidad, tiene un carácter público, donde los usuarios (en su mayoría) no suelen aportar apenas información personal y lo utilizan como plataforma para oír y ser escuchado en las redes. Lo cual lo hace idóneo para la minería de opiniones, pues la mayoría del contenido es escrito y público, y la propia plataforma provee a los desarrolladores de una API para poder acceder a la información desde los programas del proyecto.

Capítulo 2

Estado del Arte

El estado del arte se refiere al estado actual de una tecnología concreta. Siendo este el punto de partida sobre el cual cimentar el proyecto, para así evitar estudiar el mismo ámbito de la misma forma que otra gente, siendo un pilar necesario en el avance, pues sino, habría que reinventar la rueda cada vez que deseamos mejora un automóvil, por ejemplo.

Esta investigación pretende encontrar que trabajos se han hecho en lo concerniente al análisis de sentimientos, al tratamiento de la Big Data y a la obtención de información en redes sociales. Habiendo mucho recorrido en todos estos ámbitos.

2.1. Análisis de Sentimientos

Se refiere al procesamiento por parte de una máquina que sea capaz de, sin intervención humana, indicar el tono emocional que desea expresar el autor del mensaje, teniendo en cuenta diversos factores y posibles significados implícitos. Esta técnica, aún esta en fase de desarrollo, pues hay mucho que recorrer para que el análisis sea verdaderamente fiable, pues aunque intenta contemplar dobles sentidos e ironía, los resultados no son todavía aceptables en muchas ocasiones. Pero el avance es inexorable y nuevas técnicas aparecen constantemente, aunque como mucho de ese desarrollo está siendo llevado a cabo en el ámbito privado. Es sabido que la administración Obama utilizó estos análisis para hacer sondeos sobre la opinión pública para afinar mejor los mensajes de campaña y poder llegar al mayor público posible, desde entonces es solo lógico asumir que toda grande corporación empresarial o política hace usos de los análisis de sentimientos para obtener información práctica de la Big Data que tenemos en la red. Con un búsqueda en Google podemos encontrar varias empresas dedicadas a este análisis, las cuales están orientadas, en su mayoría, a grandes empresas.

BrandWatch

Plataforma de escucha e Inteligencia Social, la cual proporciona interesantes herramientas de escucha social, pudiendo hacer subdivisiones por temas dentro de la misma y luego analizar el sentimiento de cada tema por separado.

Como dato anecdótico en su página web, cuentan la historia de una empresa que sacó un anuncio. Al analizar las respuestas de sus potenciales clientes se dieron cuenta que casi todos los comentarios eran negativos debido a la música repetitiva, gracias a esto, pudieron recular y sacaron en seguida un segundo anuncio donde salían rompiendo el violín que tocaba la música, dándole así la vuelta con humor al problema y tener incluso mejores resultados que con el anuncio original. Esto es un buen ejemplo de cómo analizar las opiniones a tiempo puede resultar extremadamente beneficioso.

MeaningCloud

Esta empresa proporciona un servicio online, el cual es accesible mediante una API. Proporcionan un servicio de análisis de textos variados, no es exclusivo del análisis de sentimientos, pues también proporcionan más servicios. Permite a los usuarios empotrar análisis de textos y procesamientos semánticos en cualquier aplicación o sistema. Tienen un acceso limitado gratuito, el cual es muy interesante y el seleccionado para el proyecto debido a su facilidad de acceso, donde desde desde un programa propio, es posible acceder al servicio gracias a su API, utilizando su información como el programador disponga. Su método de análisis de sentimientos es por polaridad, con datos que varian desde muy positivo, positivo, neutro, negativo y muy negativo. Incluso puede asignar diferentes polaridades a diferentes segmentos del texto. En este proyecto, al estar trabajando con el formato *tweet* solo se tendrá en cuenta la polaridad general, pues al ser textos cortos se ha estimado que los casos donde haya más de un tópico de diferente polaridad serán desestimables.

2.2. Extraccion de datos de Twitter

La propia plataforma de Twitter, tiene una API para que cualquier desarrollador pueda acceder a sus datos de forma sencilla desde cualquier programa. El inconveniente es que no pagando, el servicio es, obviamente, mucho más limitado. De forma gratuita solo son accesibles los tweets escritos en los últimos 7 días, con un límite de tweets que se pueden descargar cada 15 minutos. Pero con esta API, hay muchas formas de acceder a la información

SocialStreams

Esta plataforma proporciona conexiones de punto a punto (end-to-end) para recolectar, pre procesar y enviar la información desde la API de Twitter al

destino de tu preferencia. Proporcionando un acceso sencillo a la información para aquellos que no quieren tener que programar, directamente seleccionas la plataforma que deseas consultar (Twitter, Reddit, Linkedn, etc.), seleccionas el formato preferente de salida (Base de datos, CSV, JSON). De esta forma, pagando por su servicio cualquiera sin conocimientos de ningún tipo de desarrollo software puede acceder a los datos.

Python Twitter Tools

Esta API para Python, disponible en Pypi, el repositorio de software oficial para aplicaciones en lenguaje Python. Proporciona una API minimalista de Twitter, una herramienta de linea de comandos para obtener y enviar tweets y un bot IRC, el cual proporciona funciones automatizadas, pudiendo anunciar por ejemplo, actualizaciones de Twitter en un canal IRC. Esta herramienta casi fue la elegida, porque la preferencia era trabajar en un entorno Python, pero al ser tan centrada en el formato twitter que no dejaba libertad para el resto del desarrollo software, llegando así a la siguiente, la que será usada en el proyecto.

Tweepy

Tweepy es una librería de Python específicamente diseñada para hacer la conexión con la API de Twitter más sencilla. Proporciona diferentes metodos RESTful (transferencia de estado representacional en castellano), los cuales son los que se usan en la web, permitiendo obtener datos o ejecutar operaciones con dichos datos, en cualquier formato, sin las abstracciones de los protocolos basados en intercambio de mensajes. Por lo que esta librería es ideal para el proyecto, ya que proporciona las herramientas de autentificación, búsqueda y streaming (escucha de tweets en tiempo real)

2.3. Almacenamiento de datos

Una vez obtenidos los datos, es necesario almacenarlos de alguna forma. Para ello existe una enorme gama de bases de datos a disposición del desarrollador.

SQL

SQL (lenguaje de consulta estructurada) es un lenguaje de dominio específico utilizado en programación, siendo su principal función la administración y la recuperación de información de bases de datos relacionales. Es actualmente el estándar del ANSI (Instituto Nacional Estadunidende de Estándares) y del ISO (Organización Internacional de Normalización). Pero a pesar de estos

estándares, la gran mayoría de códigos SQL no son portables entre diferentes bases de datos sin necesidad de ajustes.

MongoDB

MongoDB es un sistema de base de datos NoSQL orientado a documentos de código abierto. Tiene la ventaja de que en vez guardar los datos en tablas, como hacen las bases de datos relacionales, guarda estructuras de datos BSON, que son similares a JSON, haciendo mucho más sencilla la integración de los datos en la aplicación. Debido a que la información de Twitter está en formato JSON y por la maravillosa portabilidad que tiene con Python, en el proyecto guardaremos los datos que recibamos de Twitter en bases de datos MongoDB, ya que el formato de este es muy adecuado para las necesidades del proyecto.

Una librería muy útil para compatibilizarlo con Python es **Pymongo** la cual nos permite conectarnos a MongoDB gracias a una serie de funciones que permiten la compatibilidad

Capítulo 3

Análisis y Planificación

El proyecto constará de varias fases importantes a tener en cuenta, de las cuales distinguiremos de forma importante cuatro. Preparación, Desarrollo, Obtención de Información y análisis de resultados.

Preparación

Esta fase será tiempo dedicado principalmente al estudio del lenguaje de programación Python, aprender cuales son sus diferentes librerías y cómo aplicar los conocimientos de programación obtenidos durante el grado en este lenguaje el cual es la primera vez que utilizo. Además será necesario conocer como funciona la API de Twitter y la API de MeaningCloud, para poder obtener la información y luego analizarla. Repasar como funciona una base de datos MongoDB, utilizada previamente durante los años lectivos, pero en necesidad de refrescar los conocimientos.

También ha sido elegida LaTex para el desarrollo de la memoria, cuyos parámetros también habrán de ser estudiados para la correcta realización del proyecto junto con Excel y sus diferentes funciones, para poder sacar el máximo partido al proyecto.

Es importante también notar que para el proyecto se requerirá de acceso a dos APIs distintas, por lo que es necesario registrarse en Twitter Dev para obtener las claves de Twitter y en la plataforma de MeaningCloud, para poder hacer uso de su análisis con la clave que proporcionan.

Finalmente, se ha optado por la realización de un diagrama de Gantt para hacer un correcto seguimiento del proyecto.

Desarrollo

En esta fase será para hacer el desarrollo software correspondiente, crear los scripts que sean necesarios para obtener la información, para llevarla a

analizar y para procesarla, todo mientras la información obtenida como los resultados de los análisis se almacenen correctamente.

Obtención de información

Una vez desarrollados los primeros scripts, haré uso de los métodos la API de stream para escuchar en directo con la palabra o palabras claves deseadas. Este proceso podrá ser largo y controlado, para obtener una cantidad de información aceptable para el análisis y almacenar la misma en su respectiva base de datos MongoDB.

Análisis de resultados

Cuando ya disponemos de la información deseada, desde la propia base de datos MongoDB enviaremos la información a la API de MeaningCloud. La cual será procesada y la respuesta la almacenaremos doblemente, en una base de datos MongoDB por si requerimos de analizarla de nuevo y en un fichero CSV, el cual es ideal para luego poder analizar los resultados. Además en esta fase se desarrollará un script que pueda crear nubes con las palabras más utilizadas en la información descargada. También se tratará de obtener conclusiones sobre los datos obtenidos que puedan ser prácticas para una empresa. Es importante notar que en la versión gratuita de MeaningCloud solo tendremos 20000 créditos para gastar. Por cada tweet, debido a su longitud se gastará un crédito por tweet analizado.

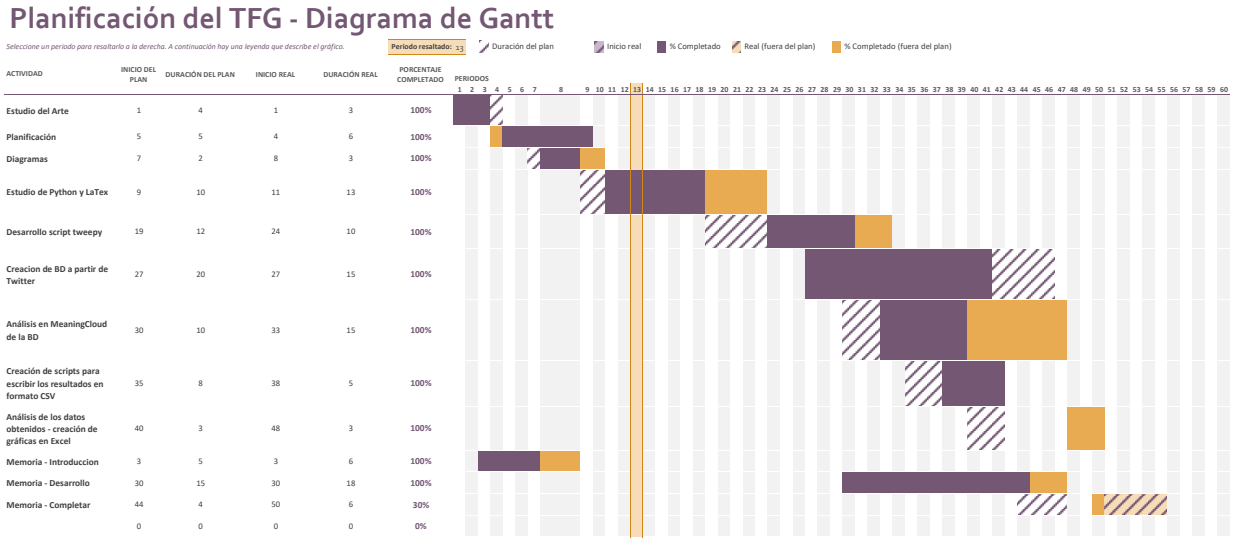


Figura 3.1: Diagrama de Gantt

Capítulo 4

Diseño

Introducir Diseño aquí

Capítulo 5

Implementación

La mayor parte del proceso de implementación estará enfocado a la creación de los scripts que sean necesarios. Primero implementaremos un programa al que llamaremos *ladrón de tweets* el cual será el encargado de obtener la información de Twitter, crear una base de datos MongoDB y almacernar los datos obtenidos en la misma.

Ladrón de Tweets

Este código hará uso de las librerías de Tweepy para acceder a la API de Twitter y las librerías de MongoDB para almacenar la información.

```
from pymongo import MongoClient
import json
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream
import datetime
```

Figura 5.1: Librerías de ladrón de tweets

Lo siguiente que hacemos es crear y conectarnos a la base de datos MongoDB, en la cuál almacenaremos toda la información que posteriormente descarguemos.

```
# Conectamos MongoDB la base de datos "TwitterStream"
connection = MongoClient('localhost', 27017)
db = connection.TwitterNetflix2108
db.tweets.create_index("id", unique=True, dropDups=True)
collection = db.tweets
```

Figura 5.2: Inicialización de base de datos MongoDB

Posteriormente es necesario declarar las variables que usaremos al usar la

API de Twitter. El idioma que buscará, las claves de acceso y las palabras claves que deseamos descargar.

```
keywords = ['Netflix' ]
# Idioma en que descargo. De momento ingles, pero trivial de cambiar si deseo
language = ['en']

# Mis claves personales de twitter. Si no puedo descargar mas tendre que usar otras cuentas para ir descargando en orden
consumer_key = "TOKEN"
consumer_secret = "TOKEN"
access_token = "TOKEN"
access_token_secret = "YOUR TOKEN"
```

Figura 5.3: Declaración de variables de acceso, búsqueda e idioma

Llegado este punto, deberemos hacer la conexión con la API de Twitter mediante las funcionalidades de Tweepy, usando las variables previamente declaradas. Con la función Stream, lo que hacemos es ponerlo en modo escucha, es decir, accederemos a los tweets que sean escritos en el tiempo de ejecución serán los que descarguemos. Debemos incluir el modo *tweet_mode=extended* el cual es necesario porque sino se usa solo se descargarán los primeros 140 caracteres del tweet, añadiendo información incompleta y por lo tanto desechable en la base de datos.

```
# Aqui se realiza la coneccion gracias a Tweepy con mis claves
if __name__ == '__main__':
    imlistening = StdOutListener()
    auth = OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)

    stream = Stream(auth, imlistening, tweet_mode='extended')
    stream.filter(track=keywords, languages=language)
```

Figura 5.4: Inicio de sesión en la API de Twitter

En la función *StdOutListener()* tendremos la parte clave del script, en donde extraeremos la información del tweet y la almacenaremos en MongoDB. Al principio del código evitaremos descargar los Retweets, ya que por experiencia, estos Retweets lo que hacen es ensuciar la base de datos, puesto que si una persona con gran influencia en redes publica un tweet que es compartido por mucha gente, no aporta nueva información sino que almacena cientos o miles de veces la misma información, invalidando en parte los resultados de su posterior análisis.

```python
class StdOutListener(StreamListener):

    def on_data(self, data):

        # Cargamos los tweets en la trash_can
        trash_can = json.loads(data)
        if  not trash_can['text'].startswith('RT'):

            # Cogemos la info que queremos del tweet para guardarlo en la base de datos    NOTA: SI QUIERO MAS O MENOS INFO EN LOS TWEETS QUE DESCARGO, AQUI TENGO QUE TOCA
            language = trash_can['lang']  # Idioma en el que esta el Tweet
            username = trash_can['user']['screen_name']  # El tweetero que escribe
            followers = trash_can['user']['followers_count']  # Los seguidores que tiene el tweetero  IMPORTANTE POR SI LUEGO QUIERO HACER ALGO CON ELLO
            tweet_id = trash_can['id_str']  # La ID del tweet en formato string
            hashtags = trash_can['entities']['hashtags']  # Hashtags que tenga el tweet
            if "extended_tweet" in trash_can:
                text = trash_can['extended_tweet']['full_text']
                print ("Entra a full text\n")
            else:
                text = trash_can['text']
            #text = trash_can['full_text']  # El tweet en si
            time_tweet = trash_can['created_at']  # Cuando se crea el tweet


            # A MongoDB no le gusta el formato del tiempo del tweet, asique lo convierto a un formato que le gusta mas y lo llamo time_for_mongo
            time_for_mongo = datetime.datetime.strptime(time_tweet, '%a %b %d %H:%M:%S +0000 %Y')


            # Junto toda la informacion en una variable tweet que sera la que guarde en la bd
            try:
                tweet = {'id':tweet_id, 'username':username, 'followers':followers, 'text':text, 'hashtags':hashtags, 'language':language, 'time_for_mongo':time_for_mongo}

                # Guardo el tweet completo en MongoDB
                collection.save(tweet)
            except:
                print("\nDuplicate Key Error\n")

            del tweet



            # Para gusto personal voy imprimiendo por pantalla los tweets que voy descargando segun los descargo en tiempo real
            print ('\n'+username + ': ' + text)
            return True
        else:
            print("\nRETWEET\n")


    def on_error(self, status):
        print status
```

Figura 5.5: Función de captura y almacenamiento de tweets

Análisis Sentimientos

Una vez existe una base de datos MongoDB hay que enviarla a analizar a MeaningCloud haciendo uso de su API. Luego de analizarla, obtendremos una información que será almacenada por partida doble, para facilitar la reutilización de la misma. Crearemos una colección diferente dentro de la base de datos MongoDB ya existente, a la que denominaremos *concepts* y a la par se creará un archivo CSV en el cual almacenaremos toda la información para facilitar su posterior análisis.

Figura 5.6: Librerías utilizadas y declaración de la BD y creación de la nueva colección concepts

También será necesario indicar las claves de acceso para la API de MeaningCloud y la dirección url de acceso a la misma.

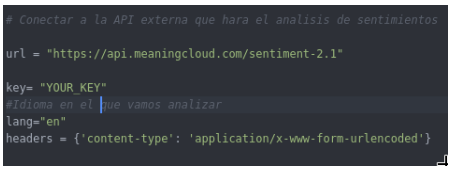


Figura 5.7: Declaración de variables necesarias para la API de MeaningCloud

El código recorrerá toda la colección *tweets* de la base de datos MongoDB, mandando únicamente el texto de los tweets a MeaningCloud, pues es la información que deberá ser analizada. Extraemos la información de utilidad de la respuesta y la almacenamos en diferentes variables. Dichas variables son:

- **Confidence:** Es el valor de fiabilidad del análisis. MeaningCloud asigna un valor de 0-100, siendo 100 lo más fiable posible a la calidad de su análisis. Sólo cogeremos los resultados de los análisis aceptables, es decir, que tengan un valor superior a 90.
- **Score_tag:** Posiblemente la variable más importante del análisis. Puesto que clasificará entre muy positivo y muy negativo el tono emocional del texto analizado. Su rango de polaridad es:
  - P+: Muy positivo
  - P: Positivo
  - NEU: Neutral
  - N: Negativo
  - N+: Muy negativo
  - NONE: Ninguno, no se le ha detectado ningún tono emocional al texto.

- **Agreement:** Si hay más de un sentimiento detectado en el texto, si estos sentimientos tienen el mismo tono emocional o no.

- **Subjectivity:** Subjetividad. Como su nombre indica, indica si el texto era objetivo o subjetivo.

- **Irony:** Indica si el texto era irónico o no. Por experiencia al usarlo esta variable será ignorada puesto que tras comprobarlo su tasa de acierto es muy baja o nula.

- **Sentimented_Entity_List:** Lista de entidades en el texto de las cuales tienen una polaridad, es decir, generan un tono emocional en el autor. Nombres de compañías de servicio, ciudades, países, nombres de usuario. Reconoce un gran rango de entidades.

- **Sentimented_Concept_List:** Lista de conceptos en el texto los cuales tienen polaridad concreta.

Finalmente, tras su análisis. Si la confianza en el análisis está en un rando aceptable, almacenamos los datos en un fichero CSV y en la nueva colección *concepts* que hemos creado dentro de la misma base de datos. Es importante notar que este análisis no admite la introducción de emojis, por lo que los tweets que contengan emojis serán devueltos con un mensaje de Error, el cual no interrumpirá el análisis y simplemente seguirá analizando el resto de la base de datos.

```
for cursor in db.tweets.find():
      tweet = cursor.get('text')
      username = cursor.get('username')
      followers = cursor.get('followers')
      id = cursor.get('tweet_id')

      print (username + ': ' + tweet)
      #payload= "key="+key+"&lang="+lang+"&txt="+txt
      payload = "key=YOURKEY&lang=en&of=json&txt= %s &txtf=plain&url=YOUR_URL_VALUE&doc=YOUR_DOC_VALUE" %(tweet)
      confidence= 0
      #Manejo excepciones por si la conexion da error que siga analizando la base de datos MongoDB
      try:
          response = requests.request("POST", url, data=payload, headers=headers)
          print(response.text)
          answer = json.loads(response.text)
          confidence = answer['confidence']
          score_tag= answer['score_tag']
          agreement= answer['agreement']
          subjectivity = answer['subjectivity']
          irony = answer['irony']
          sentimented_entity_list = answer['sentimented_entity_list']
          sentimented_concept_list = answer['sentimented_concept_list']
          quoted_tweet = '"{}"'.format(tweet)
          #Me aseguro que la confianza en el analisis este en un rango aceptable
          if int(confidence) > 90:
              try:
                  concepts = {'id': id, 'score_tag':score_tag, 'entities': sentimented_entity_list, 'concepts':sentimented_concept_list,
                              'user':username , 'followers': followers, 'text': tweet}
                  collection2.save(concepts)
                  del concepts
              except:
                    print("\n\n\n\n\n\nError al guardar en la base de datos MongoDB\n\n\n\n\n\n")

              #El parametro a es para "append", para actualizar el csv en vez de sobrescribirlo
              with open('data/YT2108.csv', 'a') as csvfile:
                  filewriter = csv.writer(csvfile, delimiter=',')
                  filewriter.writerow([username, followers, confidence, score_tag, agreement, subjectivity, irony, quoted_tweet])

          else:
              print("La confianza es demasiado bajo. Analisis no fiable \n\n\n\n")

      except ValueError:
          print("\nException: Failed request to API in meaningcloud. Wrong characters.")
```

Figura 5.8: Bucle para recorrer la BD de Mongo y analizarla

NOTA: Existe otro script llamado **analisis-desde-nombre**, el cual es casi idéntico al anteriormente descrito cuya única diferencia es que podemos analizar la base de datos desde un usuario determinado. Por lo que si hay un error, como puede ser una perdida de conexión, será tan trivial como abrir el fichero CSV, buscar el nombre del último usuario añadido al fichero y escribir dicho nombre en la comprobación del script para continuar la búsqueda sin repetir tweets y sin perder información.

WordCloud

Se ha diseñado un script, que recorrerá todas las palabras de todos los tweets del fichero CSV. Pudiendo con esta información realizar un WordCloud donde aparezcan las palabras más usadas entre todos los tweets. Para ello se hará uso de unas librerías matemáticas de uso científico. Las más importantes serán:

- NumPy: Extensión de python específica para darle mayor soporte para vectores y matrices. Constituye una librería de funciones matemáticas de alto nivel.

- Pandas: Estrechamente relacionada con la bibliotea NumPy está orientada a la manipulación y análisis de datos.

- Matplotlib: biblioteca para la generación de gráficos a partir de datos contenidos en listas o arrays, relacionada también con la extensión NumPy. Diseñada para ser similar a la utilizada en MATLAB

- PIL: Python Imaging Lybrary, es una biblioteca que añade soporte para abrir, manipular y almacenar muchos formatos de imagen distintos.

```
# Start with loading all necessary libraries
import numpy as np
import pandas as pd
from os import path
from PIL import Image
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

import matplotlib.pyplot as plt
```

Figura 5.9: Librerías utilizadas en el script de WordCloud

El código extrairá la información del fichero CSV deseado con la función *read_csv()* de la librería Pandas. Como información añadida mostrará por pantalla cuantos tweets hay de cada polaridad, desde muy positivo a muy negativo, también hará un recuento de las palabras que haya en el total de todos los tweets juntos. Los cuales se juntarán todos en una sola variable denominada text gracias a un bucle y la facilidad de manipulación de datos que ofrece Pandas.

Finalmente para este fragmento, se le introduce una imagen previamente seleccionada, la cual hará de plantilla para la posterior generación del WordCloud, es decir, en vez de usar la forma en la que aparece por defecto, usará la forma y colores de la imagen que se le introduzca. Esta imagen deberá estar en formato RPG pues gracias a Numpy se generará una máscara con la imagen transformada en un array de datos. La función array de Numpy transforma la imagen en un vector de datos comprendidos en un rango de 0-255 que contendrán la información de la imagen en un formato que el algoritmo pueda procesar.

```
df = pd.read_csv("data/JdT.csv")

score = df.groupby("Score_tag")
print(score.describe())

text = " ".join(text for text in df.Text)

print("Hay {} palabras en la combinacion de todos los tweets.".format(len(text)))

netflix_mask = np.array(Image.open("img/icono.jpg"))
```

Figura 5.10: Lectura de la imagen de plantilla y unificación de todos los tweets en una única variable

Una opción para la creación del WordCloud es la asignación de Stopwords, los cuales serán palabras que no se mostrarán en el fichero que se cree. Puesto que hay palabras que debido al formato de los tweets son propensas a aparecer mucho, podemos quitarlas para obtener un WordCloud más satisfactorio. Por ejemplo, si tengo una base de datos acerca de Netflix, es asumible que todos los tweets contendrán la palabra Netflix, sabiendo esto incluimos a Netflix entre los Stopwords para que en el fichero generado veamos las palabras más usadas para hablar de la plataforma, no la plataforma en sí.

```
# Create stopword list:
stopwords = set(STOPWORDS)
stopwords.update(["Netflix", "amp", "co", "https"])
```

Figura 5.11: Adición de Stopwords al WordCloud

Ahora queda la generación del WordCloud en sí. Llamando a la función *WordCloud()* en la que introducimos los parámetros deseados, entre los que destacan el número máximo de palabras que generaremos, la máscara que indicará la forma que debe tomar el wordcloud en sí, los stopwords que se han añadido con prioridad, el grosor de los bordes, de tenerlos, de la plantilla y el color de dichos bordes. Con la función *ImageColorGenerator()* a la cual se le añade como parámetro la máscara generada con NumPy, indicará qué colores darle a las palabras al pintar la figura. Para finalizar, con las funcionalidades de Matplotlib se pintará la imagen, habiendo que indicarle el tamaño de la figura resultante y pasarle la variable de wordcloud que contiene las palabras y por parámetro el color que deseamos que tengan dichas palabras para respetar el formato original. Guarda la figura con el nombre y formato deseado y la muestra por pantalla.

```
# Generate a word cloud image
#wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(text)
wordcloud = WordCloud(background_color="white", max_words=10000, mask=netflix_mask,
                      stopwords=stopwords, contour_width=3, contour_color='black')

wordcloud.generate(text)

#wordcloud.to_file("NetflixAll2.png")

image_colors = ImageColorGenerator(netflix_mask)
plt.figure(figsize=[7,7])
plt.imshow(wordcloud.recolor(color_func=image_colors), interpolation='bilinear')

plt.axis("off")
plt.savefig("img/JdTStark.png", format="png")
plt.show()
```

Figura 5.12: Generación de la imagen del WordCloud

Capítulo 6

Resultados

De los varios scripts implementados, se han obtenido una gran gama de resultados. Las pruebas han ido orientadas hacia los servicios de Streaming Online, debido a que hoy en día son empresas que están a la orden del día y generan gran actividad en las redes sociales. Debido al deseo de buscar una finalidad práctica para hacer estudios de mercado y al hecho de que se podían hacer dos escuchas simultáneas se ha hecho en formato de c̈arreras", es decir, a la hora de analizar HBO y Netflix, se han hecho la obtención de tweets simultáneas, para poder también juzgar que plataformas generan más tráfico en las redes sociales en los tiempos escuchados.

En total fueron bastantes las bases de datos generadas con las diferentes pruebas, el resultado que se puede apreciar en MongoDB es el siguiente:

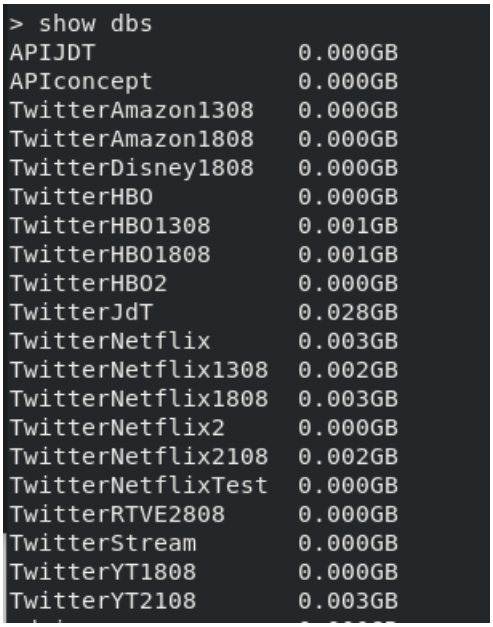
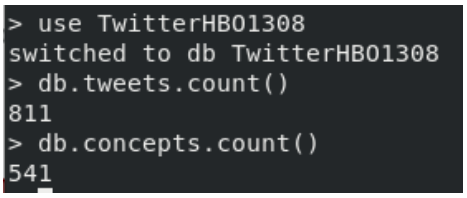


Figura 6.1: Diferentes BD generadas en MongoDB

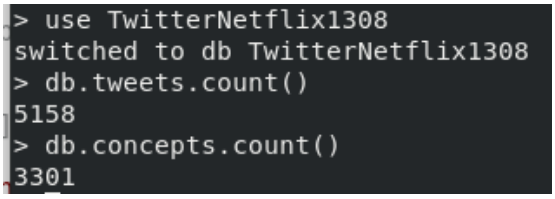
6.1. Análisis de Netflix y HBO

Al ser las dos plataformas principales y principales competencias entre sí, han sido el objeto principal de las pruebas del proyecto.

La primera competencia fue el día 13 de Agosto, de 22h a 1h. Tres horas donde fueron descargados por separados aquellos tweets que mencionaban Netflix y los que mencionaban a HBO, guardados en diferentes bases de datos.



BD de HBO1308



BD de Netflix1308

Aquí podemos apreciar dos colecciones en cada base de datos. La primera *tweets* son el total de tweets capturados por el *ladrón de tweets*, la otra *concepts* son el resultados del análisis de haberlo enviado a MeaningCloud con el *analisis-sentimientos-mongo*. Lo principal que se puede apreciar es que Netflix tiene mucha más presencia en redes que HBO, puesto que es mencionado más de 5 veces por cada vez que se menciona a HBO.

Tambiés es notable como el número de tweets que han sido analizados es mucho más bajo que los capturados. Esto es debido principalmente a que MeaningCloud no admite emojis en su análisis y siendo estos tan presentes en las redes hay una notable pérdida de información a la hora de analizarla. En este caso los tweets de HBO han tenido una tasa del 66,7 % de tweets analizables mientras que Netflix tiene una tasa del 64 %. La diferencia es casi despreciable.

Pasando a analizar los resultados del análisis de sentimientos, obtenemos lo siguiente.

Rótulos de fila	Cuenta de Score_tag	
N+	36	10,62%
N	90	26,55%
NEU	17	5,01%
P	154	45,43%
P+	42	12,39%
Total general	**339**	

Figura 6.2: Porcentaje de HBO1308

La razón por la que en el total de aparece la cifra de 339 en vez de 541 es porque 202 tweets han sido clasificados como NONE, es decir, sin ninguna polaridad emocional detectada. Del total esto correspondería al 37 % de los tweets analizados

Rótulos de fila	Cuenta de Username	
⊞ **P+**	**292**	14,37%
⊞ **P**	**920**	45,28%
⊞ **NEU**	**134**	6,59%
⊞ **N**	**527**	25,94%
⊞ **N+**	**159**	7,82%
Total general	**2032**	

Figura 6.3: Porcentaje de Netflix1308

En Netflix el 38 % de los tweets analizados no tenían polaridad. Equivalente a 1269 tweets de los 3301 analizados.

A continuación ilustraremos unos gráficos de los resultados:

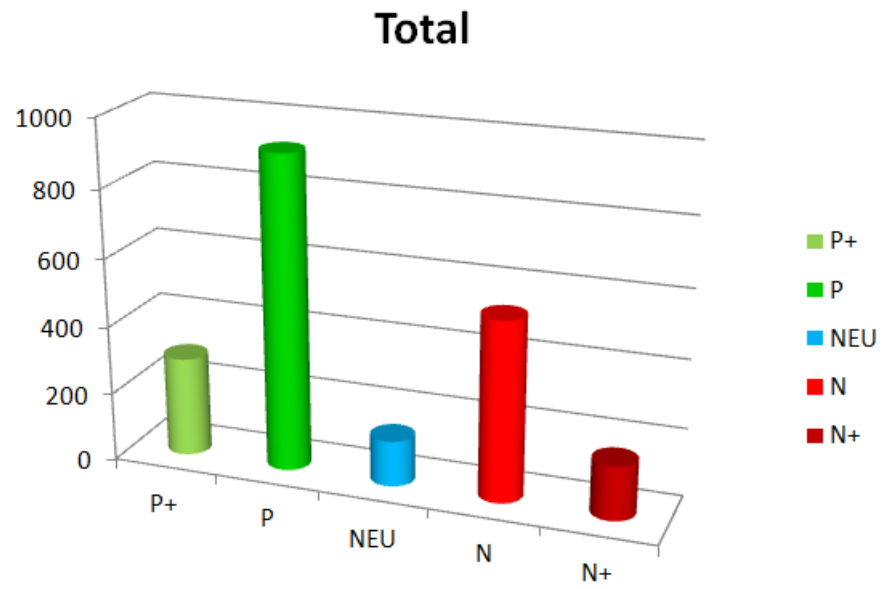


Figura 6.4: Gráfico de barras de Netflix1308

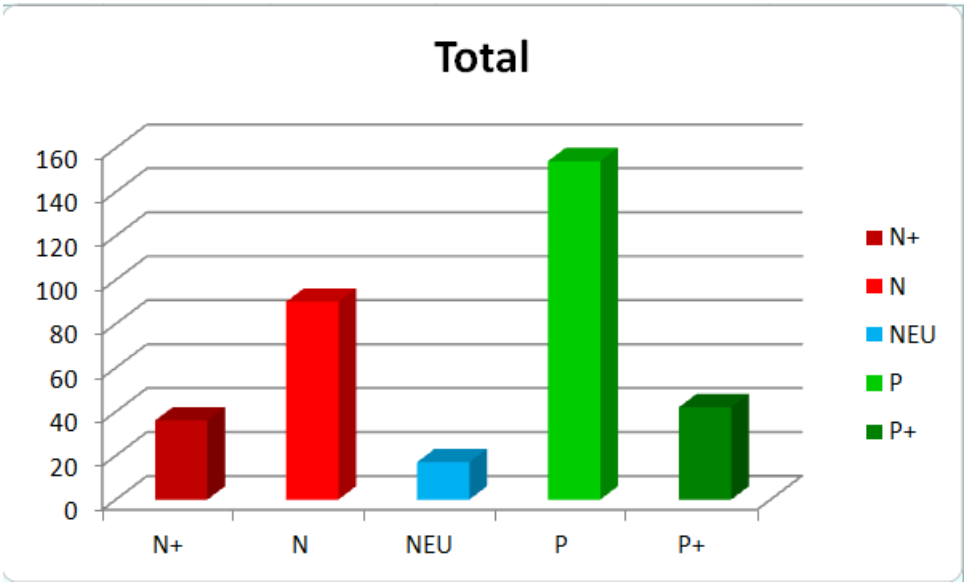


Figura 6.5: Gráfico de barras de HBO1308

Con estos resultados podemos apreciar que la mayoría de los tonos emocionales detectado son Negativos o Positivos, siendo los positivos la mayoría tanto en Netflix como en HBO.

El siguiente análisis fue realizado el 18 de Agosto. desde las 0:20 hasta las 3am. Un total de dos horas y cuarenta minutos.

```
> use TwitterHBO1808
switched to db TwitterHBO1808
> db.tweets.count()
1787
> db.concepts.count()
1285
```

BD de HBO1808

```
> use TwitterNetflix1808
switched to db TwitterNetflix1808
> db.tweets.count()
9065
> db.concepts.count()
5657
```

BD de Netflix1808

Netflix no deja lugar a dudas, vuelve a tener 5 veces más tráfico en las redes que su rival HBO. En esta ocasión, HBO ha tenido un 72 % de tweets sin emojis y Netflix se mantiene casi igual que antes con un 62 %. Analizando los resultados uno por uno se observa lo siguiente:

Rótulos de fila	Cuenta de Score_tag	
N+	265	7,87%
N	815	24,21%
NEU	237	7,04%
P	1435	42,63%
P+	614	18,24%
Total general	**3366**	

Figura 6.6: Porcentaje de Netflix1808

En Netflix a 2291 tweets, es decir, al 60 % de los analizados, no se le ha detectado tono emocional alguno.

Rótulos de fila	Cuenta de Score_tag	
N+	73	9,22%
N	220	27,78%
NEU	67	8,46%
P	357	45,08%
P+	75	9,47%
Total general	**792**	

Figura 6.7: Porcentaje de HBO1808

HBO ha tenido 493 tweets sin tono emocional, equivalente aproximadamente al 38 % del total analizado.

Las gráficas obtenidas son las siguientes:

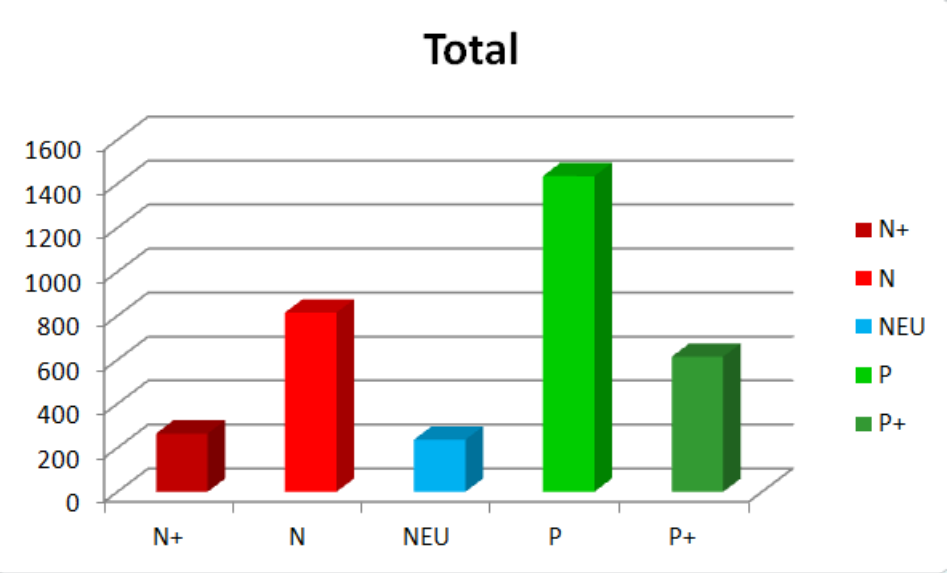


Figura 6.8: Gráfico de barras de Netflix1808

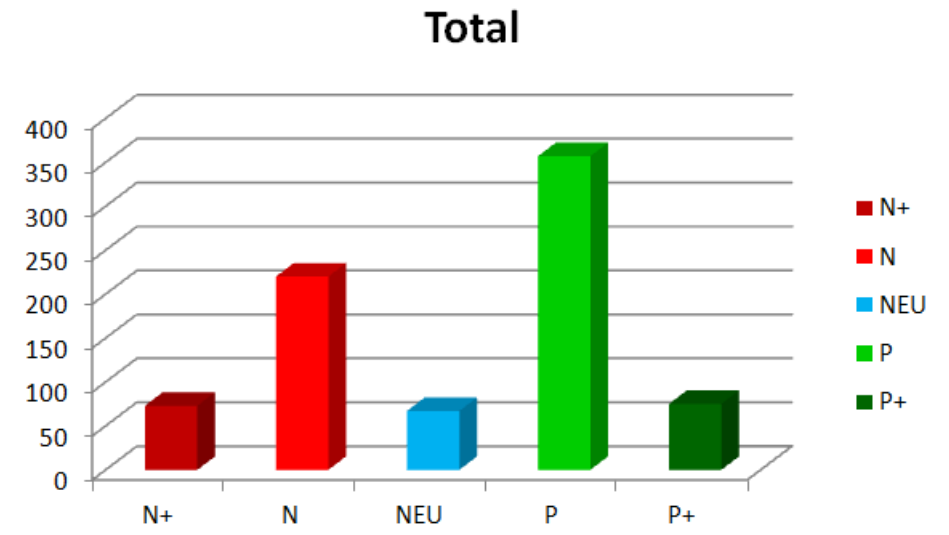


Figura 6.9: Gráfico de barras de HBO1808

Aquí se observa la misma tendencia anterior, la inmensa mayoría de la polaridad emocional es para expresar un tono positivo o negativo, pocos expresan polaridades extremas o neutras. Con la diferencia de que en HBO hay

un incremento del porcentaje de opiniones negativas con respecto a Netflix y Netflix porcentualmente recibe cerca del doble de tonos muy positivos.

Netflix vs Youtube

Al haber observado que Netflix poseía una abrumadora superioridad en cuanto a número de usuarios y actividad en las redes que HBO, se optó por hacerla competir con una plataforma mucho más grande, como puede ser YouTube. YouTube es otro formato de plataforma de Streaming Online, pero a diferencia de YouTube y HBO su contenido es gratuito y creado por los propios usuarios, aunque recientemente han sacado YouTube Premium, cuya ventaja era la de quitar los anuncios, han aprovechado para empezar a sacar contenido cinematográfico exclusivo para sus suscriptores, es decir, series y películas producidas por la compañía de Google, como llevan muchos años haciendo Netflix y HBO.

Se les analizó el 21 de Agosto, de las 16h hasta las 17:30h. Una hora y media de escucha.

```
> use TwitterNetflix2108
switched to db TwitterNetflix2108
> db.tweets.count()
3602
> db.concepts.count()
2373
```

BD de Netflix2108

```
> use TwitterYT2108
switched to db TwitterYT2108
> db.tweets.count()
6940
> db.concepts.count()
4913
```

BD de YouTube2108

Es apreciable, que siempre hay un pez más grande. YouTube en tan solo hora y media es mencionada el doble de veces que Netflix. Netflix mantiene su línea de contenido sin emojis, un 65,8 % de todos los tweets son analizables, mientras que YouTube se mantiene superior, con un 70 % de éxito en los tweets que se analizan correctamente.

Rótulos de fila	Cuenta de Score_tag	
N+	96	6,71%
N	358	25,02%
NEU	102	7,13%
P	646	45,14%
P+	229	16,00%
Total general	**1431**	

Figura 6.10: Porcentaje de Netflix2108

En Netflix ha habido 942 tweets sin tono emocional, lo que equivale al 39,69 % de los analizados.

Rótulos de fila	Cuenta de Score_tad	
N+	238	8,38%
N	773	27,22%
NEU	135	4,75%
P	1327	46,73%
P+	367	12,92%
Total general	**2840**	

Figura 6.11: Porcentaje de YouTube2108

En YouTube hubo 2073 tweets carentes de tono emocional, 42,12 % del total.

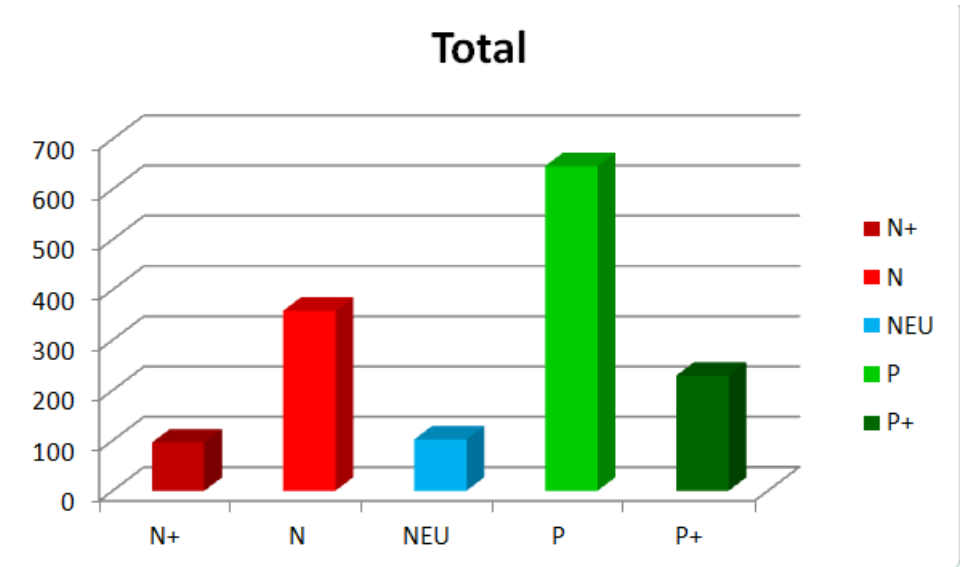


Figura 6.12: Gráfico de barras de Netflix2108

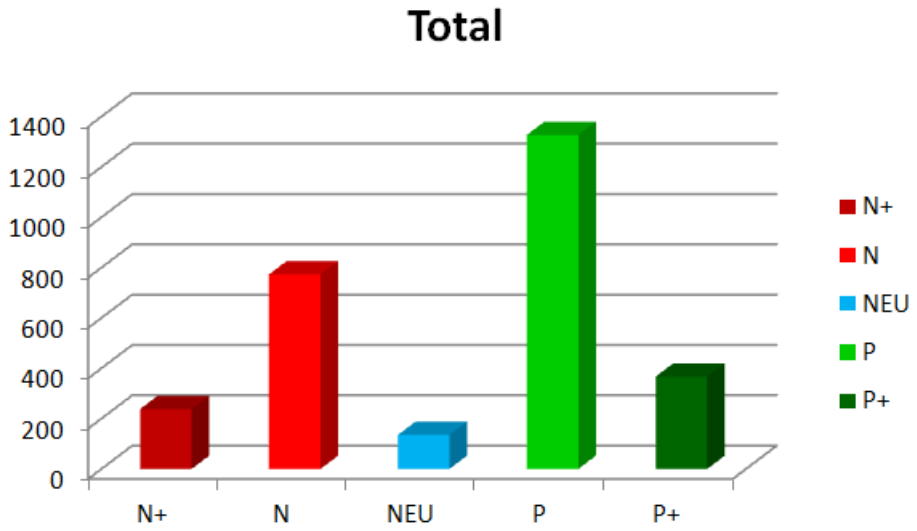


Figura 6.13: Gráfico de barras de YouTube2108

En cuanto a las gráficas se refieren, no hay diferencias notables entre las dos plataformas. Se mantiene el tono positivo a la cabeza con el negativo detrás. Aunque levemente, en YouTube hay un mayor porcentaje de tonos extremos, es decir, hay más tonos muy positivos y tonos muy negativos que en Netflix.

Análisis del total analizado

Juntando todos los tweets analizados en ficheros únicos gracias al script *unificador-csv* para poder analizar los resultados completos.

Netflix:

Rótulos de fila	Cuenta de Score_tag	
N	1700	15,01%
N+	520	4,59%
NEU	473	4,18%
NONE	4500	39,72%
P	3001	26,49%
P+	1135	10,02%
Total general	**11329**	

Porcentajes de Netflix total

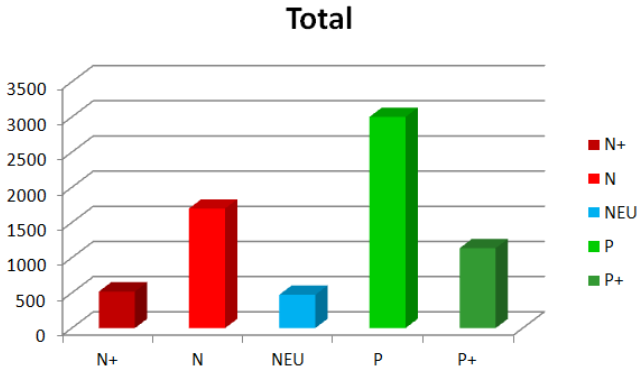


Gráfico de barras de Netflix total

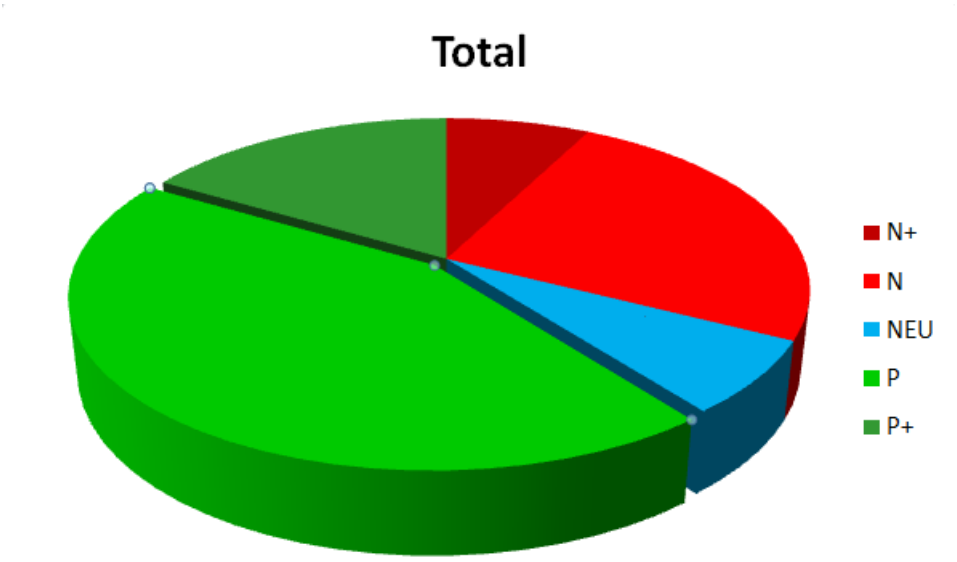


Figura 6.14: Piechart de los resultados de Netflix

Es claramente apreciable que la gran mayoría de los comentarios de Netflix son positivos y muy positivos. Hay más comentario negativos que muy positivos, pero la diferencia es mínima y siendo solo el 39 % del total positivo es superior al 19 % que suman los tweets negativos y muy negativos.

HBO:

Rótulos de fila	Cuenta de Score_tag	
N	320	17,52%
N+	116	6,35%
NEU	90	4,93%
NONE	630	34,50%
P	547	29,96%
P+	123	6,74%
Total general	**1826**	

Porcentajes de HBO total

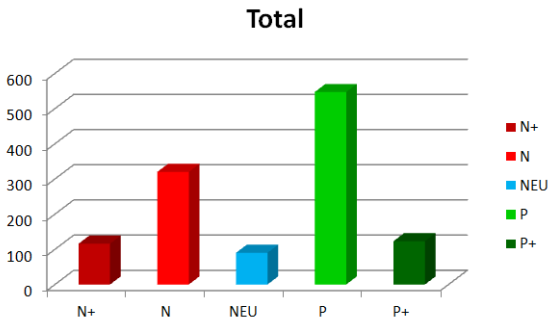


Gráfico de barras de HBO total

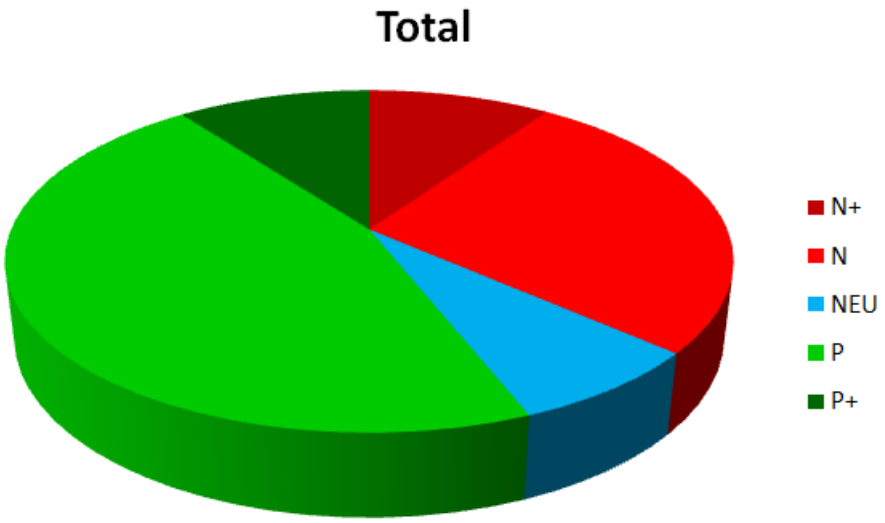


Figura 6.15: Piechart de los resultados de HBO

La tendencia es similar a la de Netflix, puesto que la mayoría de comentarios son positivos y se tienden a evitar los extremos y los neutros. Pero en HBO la diferencia entre positivos y negativos es menor, es decir, hay una mayor porcentaje de usuarios que escriben comentarios negativos al hablar de HBO.

Disney+ y AmazonVideos

También, como dato anecdótico, se intentó hacer competir la repercusión en redes de la plataforma que va a sacar Disney al mercado, Disney+, y la plataforma de Amazon, AmazonVideos. Pero los resultados fueron poco alentadores, con apenas 150 tweets de Disney+ publicados durante la captación y tan solo 3 tweets de AmazonVideos en el mismo espacio, cabe notar que los tweets de Amazon fueron todos publicados por un mismo usuario repitiendo un mismo mensaje.

WordClouds

Finalizada esta etapa de análisis de resultados, pasamos a los resultados del siguiente script relevante del proyecto. Durante esta fase se han ido haciendo diferentes pruebas con diferentes bases de datos y comprobaciones de como hacer los wordclouds mejor.

Inicialmente los Wordclouds se veían así:

Figura 6.16: Primer WordCloud de Netflix

El primer fallo notable era que la palabra Netflix aparecía muy repetida, lo cual es obvio, pues Netflix era la palabra clave utilizada para descargar los tweets, es decir, todos los tweets tenían la palabra Netflix presente. Además, un gran numero de ellos incluye enlaces, al empezar todos los enlaces por https el algoritmo detectó esta palabra y sale repetidas veces. Además aparecen ämp¨, utilizado por caracteres especiales y c̈oq̈ue aparece en los enlaces. Implementando el uso de stopwords y cambiando el fondo a blanco por limpieza visual el resultado fue el siguiente:

Figura 6.17: Segundo WordCloud de Netflix

En esta segunda versión, se aumenta el número de palabras representadas y el uso de StopWords para limpiar los resultados. Los resultados obtenidos son los que nos podíamos esperar del análisis de un wordcloud de Netflix, palabras como watch, movie, season entre otras están presentes. Es notable como leyendo los resultados se puede extrapolar como la mayoría de usuarios lo utiliza para hablar de series y programas que le gustan: love, good, watch, now. Y como también hay otros usuarios que comentan para recibir recomendaciones: something watch , please, Looking.

Pero este formato no llegaba del todo a cumplir unas expectativas de estética visual. Por lo que se siguió mejorando el script. El objetivo era utilizar imágenes como plantilla a la hora de dibujar los wordclouds. Para los primeros intentos se utilizó la siguiente imagen:

Figura 6.18: Logo de Netflix

La cual daba como resultado a wordclouds como este:

Figura 6.19: Tercer WordCloud de Netflix

Aunque tenían cierta similitud, este no era el resultado deseado. Se podía apreciar la diferencia entre los colores, el centro rojo y los bordes negros, pero no la silueta real del logo.

La solución se puedo encontrar prácticamente sola al replicar esta misma técnica con los datos de HBO, al utilizar el siguiente logotipo:

Figura 6.20: Logo de HBO

El cual daba como resultado el siguiente WordCloud:

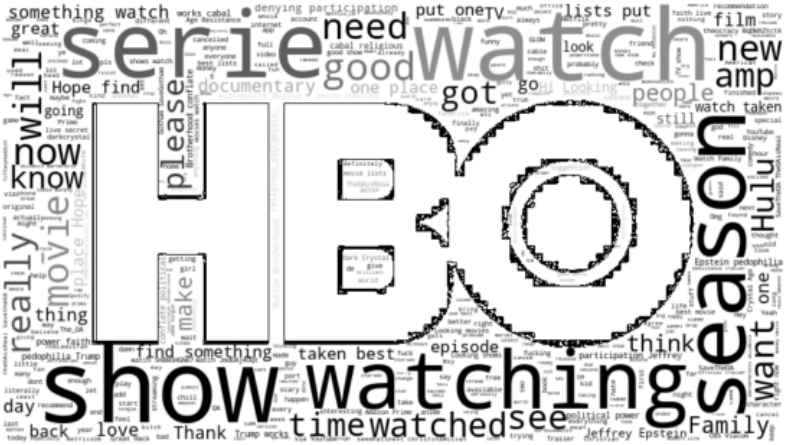


Figura 6.21: WordCloud de HBO

La diferencia principal era que este logo tenía unos bordes blancos mejor delimitados, los cuales el algoritmo podía detectar y así darle la forma deseada al dibujo para que quede mejor. Las palabras en sí no se aprecia gran diferencia de las de Netflix, al ser un servicio parecido las palabras para hablar del mismo, inevitablemente son también parecidos. Con el detalle de que aparece mencionada la plataforma Hulu, la cual tras una breve búsqueda en Google podemos averiguar que en Estados Unidos esta plataforma ofrece los servicios de HBO de la misma forma que Movistar ofrece Netflix en España.

Para aplicar esta solución a Netflix, se buscó otro logo que tuviese unas delimitaciones blancas que pudiese utilizar el algoritmo.

Figura 6.22: Segundo logo de Netflix

Desde el cual se dibujaba el siguiente WordCloud:

Figura 6.23: Wordcloud final de Netflix

Juego de Tronos

Como añadido, durante la semana del final del último capítulo de la serie Juego de Tronos hubo una versión previa del *ladrón de tweets*, capturó más de 100mil tweets solo en dos días, el problema es que no tenía un filtro para evitar los Retweets, por lo que la inmensa mayoría de la información es el mismo tweet repetido una y otra vez. Además no estaba gestionado el modo extendido para capturar tweets de más de 140 caracteres. Por lo que su utilidad real para análisis de sentimientos es escasa, pero aún así se le ha hecho un análisis de sentimientos parcial, pues el número de tweets era abrumador y un wordcloud.

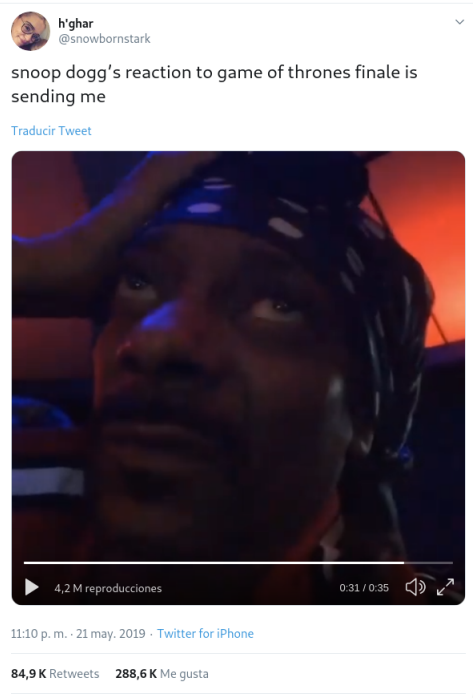
El total de palabras analizadas es de 19.809.698, las cuales generan el siguiente wordcloud:

Figura 6.24: Wordcloud de Juego de Tronos

Una de las cosas que más llama la atención de este wordcloud fue la gran presencia de Ävatar the Last Airbenderÿ de Snoop Dogg¨. Esto se debe a dos tweets diferentes publicados mientras se realizaba la captura de tweets, los

cuales obtuvieron una enorme cantidad de Retweets y se almacenaron en la base de datos MongoDB miles de veces.

Tweet de Avatar Last Airbender



Tweet sobre la reacción de Snoop Dogg

Figura 6.25: Tweets que tuvieron una alta tasa de Retweets y ensuciaron la base de datos

Y la forma que toman los 3500 tweets llevados a analizar a MeaningCloud es la siguiente:

Score_tag	Count - Score_tag	
N	253	7,13 %
N+	286	8,07 %
NEU	19	0,54 %
NONE	2580	72,76 %
P	379	10,69 %
P+	29	0,82 %
Total Result	**3546**	

Porcentajes de Juego de Tronos

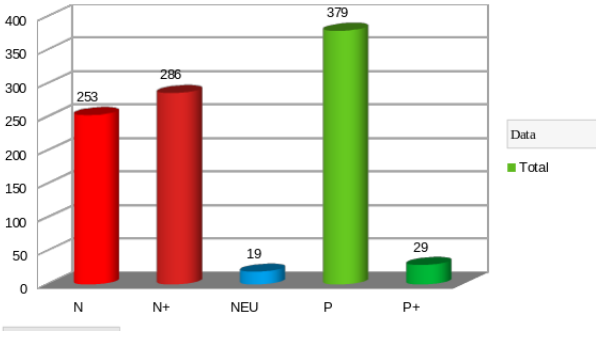


Gráfico de barras de Juego de Tronos

Figura 6.26: Análisis de sentimientos de Juego de Tronos

Lo más destacable es que los tweets muy negativos son muy altos, y la suma de ambos negativos superan con creces a la suma de las polaridades positivas, dejando clara que la tendencia en las redes es bastante negativa al respecto de Juego de Tronos. Lo cual se hizo patente fuera del proyecto, pues hubo un enorme descontento en las bases de los fans al respecto del final de la serie. Otro dato relevante es que los tweets sin tono emocional son muchos, esto se debe principalmente a que muchos de los tweets analizados estaban incompletos debido al modo de tweet extended.

Capítulo 7

Conclusiones

Durante la realización de este trabajo, he llegado a la conclusión de lo útil y completo que es el lenguaje de programación Python, como gracias a los aportes de la comunidad, se han desarrollado librerías muy potentes que permiten realizar tareas muy complejas con cierta sencillez. Esto es posible gracias a la distribución de software libre que nos permite no tener que reinventar la rueda cada vez que deseamos hacer un proyecto.

Por ello pongo a disposición de quién lo desee este proyecto, bajo una licencia GNU General Public License, la cual es abierta para el que quiera usarlo. El proyecto se puede encontar en el repositorio Github:

https://github.com/mikykeane/TFG/

También me ha hecho valorar aún más la importancia de la Big Data. De cómo vivimos en un mundo cada vez más público, dónde pagamos por servicios online con nuestra información, la cual será comprada por grandes empresas para analizarla y exprimirla todo lo posible. Esto abre un mundo de posibilidades, algunas excitantes y otras aterradoras, pues el progreso en sí no es ni bueno ni malo, solo el uso que hagamos del mismo puede estar sujeto a la moralidad.