Sprint 1: Investigación

Jorge Martín Arévalo
Guillermo Novillo Díaz

Índice

1. Tecnologías

El objetivo de esta aplicación es desarrollar una plataforma en la que los usuarios puedan subir los datos que Twitter genera, obteniendo una interpretación mucho más profunda de la que nos ofrece la aplicación.

Para ello, haremos uso del lenguaje Python (empleado para el análisis de los datos). En este apartado vamos a observar para cada una de las principales funcionalidades que hemos identificado dentro de nuestra aplicación las tecnologías predominantes.

1.1. Mostrar la información

Tenemos dos opciones en función del tipo de aplicación que vamos a hacer (app web o bien un ejecutable en local):

- **Aplicación web:** los principales frameworks que se emplean en Python para el desarrollo de aplicaciones web son Django y Flask. Existen frameworks más concretos para el desarrollo de web analíticas con gráficos, Dash y Streamlit.
  - *Django:* se trata de un marco de trabajo que ofrece una gran cantidad de funciones predefinidas y posee una base de datos interna. Por otro lado, presenta muy poca flexibilidad de cara al desarrollo y además su uso puede resultar tosco y complejo.
  - *Flask:* mucho más simple y flexible que Django, pero únicamente implementa la parte del front (a diferencia de Django que también hace Backend).
  - *Dash:* es una suma de Flask (aporta las funcionalidad web del servidor), React.js (renderiza las interfaces de usuario en la web) y Plotly.js (genera los gráficos). Se pueden crear gráficos interactivos y proporciona MUCHÍSIMOS componentes HTML ya creados para no tener que picar todo el código html y el js. También se puede utilizar bootstrap para los estilos!! Todo el html se programa en python. Se ejecuta como un script de python que abre una página web local. Hay mucha documentación en YouTube y la propia de Dash parece muy clara. El video lo explica muy bien.
  - *Streamlit:* mismo funcionamiento que la anterior pero presenta componentes muy limitados para construir las interfaces de usuario y no permite hacer visualizaciones complejas. Además, tiene menos posibilidades de despliegue que Dash. Como punto fuerte, la curva de aprendizaje es menor. Ver cuadro comparativo en la bibliografía.
- **Ejecutable en local:** si optamos por esto, tendremos que presentar el script usando GUIs para poder mostrar toda la información que se está usando. Los principales frameworks van a ser PyQt5 y Tkinter.
  - *PyQt5:* se trata de una marco de trabajo compuesto por directorios (como se puede ver en proyectos Java). Para ello vamos a definir (usando funciones) cómo vamos a crear las componentes.
  - *Tkinter:* anteriormente estaba incluida con Python pero ahora tienes que importarla instalando la librería. El funcionamiento se realiza creando instancias de los distintos objetos (botones, ventanas, ...). A simple vista parece un poco más complejo que el anterior.

1.2. Manipular la información

Tenemos dos opciones para realizar las operaciones sobre los datasets en los que tenemos la información: Pandas y PySpark.

- **Pandas:** Dentro de las librerías que ofrece Python podemos encontrar Pandas. Esta librería es una de las más comunes y utilizadas en el ámbito del análisis de datos, debido a las herramientas que ofrece así como la compatibilidad con librerías gráficas (como Matplotlib).

  Además, los datos que tenemos como objetivo analizar se encuentran en formato JSON, uno de los tipos que se encuentra soportado en esta librería y para el que además ofrece la posibilidad de crear DataFrames, una estructura de datos compleja con la que poder realizar una gran cantidad de operaciones para extraer conclusiones acerca de los datos contenidos.

He consultado algunos blogs de opiniones acerca de Pandas y la mayoría se ponen de acuerdo en que sí es posible utilizarlo para procesar Big Data, pero que existen algunas tareas, como ordenaciones, joins o funciones de agregación que pueden dar problemas de rendimiento para grandes cantidades de datos.

Otro de los principales inconvenientes que tiene es que no permite paralelizar las tareas (PySpark sí que dejaba) de cara a optimizarlo y a tener un mejor rendimiento.

- **PySpark:** PySpark está diseñado para manejar procesamiento de datos que con Pandas no sería factible debido a restricciones de memoria, como algoritmos iterativos o machine learning sobre datasets muy grandes.

  Existen técnicas para cargar datos enormes en Pandas sin sobrecargar la memoria. Se puede utilizar el atributo chunksize o dividir el dataset en archivos más pequeños e ir leyéndolos poco a poco. Además, existe mucha documentación sobre optimización de memoria en esta librería. Resumen video: 1 - 5 GB Pandas, 5 - 30GB Pandas con chunksize, 30 - 100GB Dask, 100GB en adelante Spark.

1.3. Análisis de sentimientos

La principal librería empleada en este apartado es NTLK, una librería orientada al procesamiento del lenguaje natural y que a su vez tiene implementadas una serie de funciones que te permiten evaluar la positividad/neutralidad/negatividad de un fragmento de texto (en nuestro caso sería de un tweet).

También hay otras librerías de procesamiento del lenguaje natural que pueden resultar un poco más complejas, como scikit learn o gensim (para los word embedding).

1.4. Decisión final de las tecnologías

Para la elección de la librería de manipulación del dataset nos basamos en los siguientes criterios:

1. **Tamaño del dataset:** usamos Pandas porque no esperamos datasets de más de 10GB. Además vamos a realizar una primera fase de data cleanning la cual reduciría el tamaño. Pensamos usar técnicas de optimización de memoria.
2. **Complejidad de la tarea:** usamos Pandas porque no esperamos datasets de más de 10GB. Además vamos a realizar una primera fase de data cleanning la cual reduciría el tamaño. Pensamos usar técnicas de optimización de memoria.
3. **Curva de aprendizaje:** usamos Pandas porque es más sencilla la manipulación de los dataframes al ser mutables.
4. **Recursos disponibles:** PySpark requiere un clúster o un sistema distribuido para ejecutarse, por lo que necesitará acceso a la infraestructura y los recursos adecuados. Si no tiene acceso a estos recursos, pandas es una buena opción.

Para la elección de la librería de visualización del dataset hemos decidido que finalmente vamos a usar Dash.

2. Estado del arte

Algunas de las principales herramientas que hemos identificado que realizan un análisis de los datos de las redes sociales (concretamente Twitter) son:

- **Awario:** se trata de una herramienta que permite el análisis de las menciones que se han realizado al usuario en las distintas redes sociales que tiene asociadas a un mismo correo electrónico.
- **TweetDeck:** aplicación nativa de Twitter en la que se muestran las principales fuentes de información (notificaciones, publicaciones de tus seguidores y tus publicaciones), permitiendo consultar de forma mucho más visual la información. Actualmente es únicamente para usuarios de pago y permite la creación de actividad desde ahí.
- **TweetBinder:** aplicación de análisis de redes sociales que permite a los usuarios conocer la importancia/relevancia de algún hashtag concreto o alguna palabra clave. Está muy orientada al mundo de la empresa, para que puedan conocer si sus campañas funcionan.

- **Tweetonomy:** otra de las aplicaciones que se encuentran orientadas a la empresa. Permite la obtención de un report en el que aparecen las menciones, hashtags y palabras clave que se han utilizado. Tiene también un apartado en el que se puede ver un histórico de los nuevos seguidores.
- **Sígueme:** tiene una funcionalidad bastante similar a lo que estamos buscando. Se trata de un dashboard que a partir de tu nombre de usuario te muestra las palabras clave que más has utilizado en tus tweets, así como los hashtags que más has compartido. También te muestra un histórico de todas las menciones que has realizado.
- **Tweepsmap: (ahora Fedica)** muestra estadísticas demográficas y permite mapear a los seguidores. Comprender en qué están interesados sus contactos influyentes en Twitter. Conocer el sentimiento de sus seguidores (alegría, tristeza y más). Es muy completa, utiliza IA y la API de Twitter en modo pago.
- **Followerwonk:** adquirido por Fedica por los cambios en la API de Twitter.
- **Chirpty:** círculo con los avatares de las cuentas con las que más interactúas, divido en tres: el primero de 8 contactos, el segundo de 16 y el tercero de 26. Como la API de twitter gratuita solo es de escritura, el proyecto ya no funciona. Pero... El proyecto es público en github y explica cómo crear la imagen y el algoritmo para calcular el grado de interacción con la persona en función de los likes, rt y respuestas.

Cabe destacar que la gran mayoría de estas aplicaciones son de pago (si es que siguen en funcionamiento), por lo que la información que tenemos de ellas es muy limitafa.

Algunas de nuestras ideas:

- Con el atributo locationHistory (Location history associated with the account based on activity from the last 60 days) del archivo personalization.js realizar un mapa con los diferentes puntos marcados.
- Gráfico circular de interacciones en función de los likes, rts, respuestas, menciones, etc. Mirar el algoritmo de los de Chirpty y cómo construyen la imagen en forma de círculo.

3. Bibliografía

- Cuadro comparativo Streamlit vs Dash
- Empezando un proyecto con Dash desde cero
- Creación de una interfaz sencilla usando PyQt5
- Creación de un dataframe en Pandas a partir de un JSON
- Pandas vs PySpark
- Uso de la librería NLTK para el análisis de sentimientos
- Limitación de Pandas con conjuntos de datos muy grandes
- Algunas de las principales aplicaciones que hemos mencionado y sus interfaces
- Chirpty Web del Proyecto