Universidad Autónoma Metropolitana – Azcapotzalco

División de Ciencias Básicas e Ingeniería

Licenciatura en Ingeniería en Computación

Reporte final del proyecto de Integración

Aplicación web para minería de comentarios en twitter

Proyecto Tecnológico

Alumno:
Saucedo Vargas Ezra
210200913

Asesor:

Dr. José Alejandro Reyes Ortiz
Departamento de Sistemas

Trimestre 2017 Invierno

Fecha de entrega

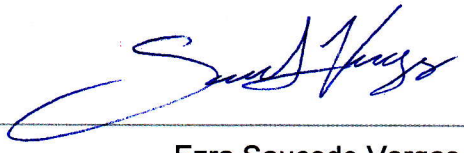
25 de abril de 2017

Declaratoria

Yo, Dr. José Alejandro Reyes Ortiz, declaro que aprobé el contenido del presente reporte de Proyecto de Integración y doy autorización para su publicación en la Biblioteca Digital, así como en el repositorio Institucional de la UAM Azcapotzalco.

Dr. José Alejandro Reyes Ortiz

Yo, Ezra Saucedo Vargas, doy mi autorización a la Coordinación de Servicios de Información de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, para publicar el presente documento en la Biblioteca Digital, así como el Repositorio Institucional de la UAM-Azcapotzalco.



Ezra Saucedo Vargas

Resumen

El análisis de los datos obtenidos en un sistema nos ayuda a poder entender mejor el comportamiento del mismo, nos permite estudiarlos, e incluso poder extraer información esencial para el mejoramiento, prevención, toma de decisiones o simplemente tener una visión más completa del panorama.

Dado que los datos se encuentran hoy die en todas las cosas, desde bases de datos, hasta el internet de las cosas, en este proyecto de integración se propone diseñar una aplicación web que realice minería de datos específicamente a los comentarios de twitter en streaming, de tal manera que se podrá saber en casi tiempo real cual es el sentimiento de los tuits más resientes.

Para ello, se usará NodeJS como freamwork de desarrollo de la aplicación web, el cual es puro JS, también se usará HTML5 y CSS3.

La finalidad de este proyecto de integración es poder clasificar y graficar la polaridad del sentimiento (positivo, neutro y negativo) de los comentarios de la red social twitter, y saber, así como se expresa el público sobre determinado objeto, servicio, producto o persona, en tiempo casi real.

Índice

Indice de Figuras:

1. Introducción

La información y el conocimiento siempre han sido sinónimo de poder y de ventaja sobre aquellos que desconocen o ignoran algo. Hoy día la información se encuentra en los millones de datos que procesamos desde ingresar a un portal en internet, hasta el comunicarnos con otros utilizando dispositivos inteligentes. Haciendo de los datos una gran mina de información.

En la actualidad la manipulación y procesamientos e interpretación de datos no son un tema trivial, podemos hallar gran complejidad en ello, desde el punto de vista de la seguridad como: almacenamiento, manipulación, transporte e interpretación, entre otros. Sin embargo, hoy día existen técnicas y algoritmos que nos permiten no solo interpretar los datos que vemos sino aquellos que no vemos de manera explícita, al proceso de obtención de esta información se le conoce como minería de datos.

Mediante la minería de datos podemos procesar la información de grandes cantidades de datos, en busca de patrones y tendencias. Y no existe mayor cantidad de datos, que se generan día a día, que en las redes sociales.

Las redes sociales hoy día son el medio por el cual millones de usuarios alrededor del mundo publican libremente opiniones y comentarios sobre temas sociales, culturales, políticos, científicos, tecnológicos o simplemente gustos o aceptación por algún producto o servicio.

Millones de datos son los que se procesan en las redes sociales cada minuto, y la cantidad de información que se esconde en esos datos es tan valiosa, que empresas y gobiernos la buscan. Con finalidad de saber que impacto tiene su producto, marca, o presencia ante el público o simplemente para saber las tendencias o patrones que se van a generar.

Para tal fin este trabajo de integración se implementa una aplicación web que pueda realizar minería de datos en los comentarios de twitter ya que es una de las principales redes sociales con mayor impacto. Esta aplicación web permite la obtención de los tweets más recientes mediante el API Streaming de twitter, de tal manera que la información es de los más reciente, y a la vez que se va obteniendo los twitteres, la aplicación web, empieza a analizarlos y a asignarles una polaridad, dependiendo el contenido de estos.

2. Antecedentes

En esta sección se muestran algunos trabajos relacionados con este proyecto.

2.1 Proyectos de integración o terminales:

Sistema visualizador de información extraída de modelos ontológicos del dominio académico basado en web [1]

El objetivo de este proyecto es similar en que mediante el desarrollo de una aplicación web se quiere mostrar gráficamente al usuario el contenido de un modelo ontológico, con el fin de estructurar mejor los conceptos y sus relaciones entre sí. Ya que un resultado grafico bien diseñado puede ser mejor interpretado por el usuario.

Aplicación web de administración de horarios para estudiantes [2]

El propósito de este proyecto fue el desarrollo de una aplicación web que administraba los horarios para los estudiantes, ayudándoles a recordar los eventos y actividades agendadas para cada día. Para el desarrollo de esta aplicación web se utilizó el modelo vista controlador (MVC), mismo modelo con el que se desarrollara la aplicación web de esta propuesta. En el cual establece la separación entre el código necesario para representar los modelos, las vistas y los controladores de la aplicación.

2.2 Tesis

Estándares en el diseño de los sitios web de instituciones educativas de nivel superior [3]

Esta tesis identifica las necesidades y los retos que enfrentan los sitios web de instituciones educativas de nivel superior, para lograr que los sitios web sean interesantes para los usuarios. Como se menciona en esta tesis el conocer las particularidades y las características de los elementos que conformaran esta propuesta ayudara a generar un producto de calidad y de mejor funcionalidad.

Herramienta para el pre procesamiento de tweets con base en búsqueda por tópico [4]

Esta tesis propone no solo analizar los datos o tuits por persona, por ubicación o por fecha de publicación, si no que propone analizarlos por tema y realizar pre procesamiento mediante técnicas y algoritmos ya definidos para la realización de minería de datos. Esto permitirá obtener mejores resultados en las búsquedas para encontrar un tema. De la misma manera esta propuesta utilizara algoritmos de minería de datos ya definidos y la utilización del API de Twitter para la obtención y análisis de los datos.

2.3 Software

SWB Social [5]

Es una herramienta que utilizando la tecnología de semántica e implementando el análisis, monitoreo de mensajes y sentimientos puede identificar cuando un comentario es positivo, negativo o neutro. Además, esta herramienta permite que la aplicación aprenda a identificar los mensajes y a mejorar su desempeño en clasificación y atención. Pero a diferencia de la propuesta es una herramienta de uso comercial, por lo que tiene un costo su utilización.

Keyhole [6]

Es una aplicación web que se asemeja mucho a esta propuesta, el diseño es claro, evalúa y Gráfica los tuits, pero al mismo tiempo brinda resultados de la red social Instagram, llegando a ser un poco compleja al interpretar los resultados y aparte que es una

herramienta de uso limitado solo por 3 días, después uno tiene que registrarse y pagar para continuar utilizándola. En este sentido la propuesta que realizamos será más sencilla y clara al mostrar los resultados.

3. Justificación

El creciente aumento de usuarios en las redes sociales ha generado nuevos campos de estudio y de interés, ya que millones de usuarios comentan y publican a cada segundo, generando una cantidad inmensa de datos. Prácticamente cualquier persona puede comentar, publicar, expresar lo que siente, lo que piensa, su aceptación o rechazo hacia una persona, una marca, un servicio, un equipo o incluso hacia una ideología. Toda esta cantidad de datos que se generan a cada instante dice más de lo que a simple vista podemos observar.

Los datos, hoy día son como el oro, hay que minarlos, y entonces podremos encontrar el verdadero valor que hay en ellos. La mayoría de los análisis convencionales de datos solo aplican analítica descriptiva que es la que como su nombre nos dice describe los datos, nos da la información que ya está implícita en ellos. Pero el verdadero reto es encontrar en esos mismos datos la información que no está de manera implícita, y esta información se obtiene mediante minería de datos, máquinas de aprendizaje, modelos de redes neuronales y big data.

Para obtener esta información oculta en los datos, la cual es muy valiosa, existen herramientas que, mediante el uso de algoritmos, analítica predictiva y otras técnicas avanzadas de minera de datos, logran obtener esta información oculta en millones de datos, generando patrones o tendencias. Algunos ejemplos de la información que se podrían encontrar aplicando minería de datos a comentarios de twitter van desde identificar el sentimiento del twittee, el sexo del autor, la edad del autor, el nivel educativo del autor, etc. Entre otros. Esta información pudiera parecer poco valiosa a simple vista, pero cuando se pone en el marco y contexto adecuado puede ser una gran herramienta, para las empresas, organizaciones privadas, sector educativo, sector salud, e incluso para el gobierno.

Dichas herramientas suelen interpretar la información obtenida y presentarla al usuario. Sin embargo, dichas herramientas suelen ser de uso comercial, por lo que son costosas y en ocasiones un poco complejas de operar, también la baja disponibilidad de este tipo de herramientas para el idioma español de México, hace que sea difícil o ineficiente el uso de estas herramientas.

A partir de la importancia de analizar la información obtenida en los datos, específicamente de los tuits o comentarios de twitter. Es por ello que en esta propuesta se plantea el diseño y desarrollo de una aplicación web, la cual facilitara la interpretación y la visualización de la información obtenida del análisis de los tuits, mediante gráficas. Específicamente para tuits en español de México.

4. Objetivos

4.1 Objetivo General:

Diseñar y desarrollar una aplicación web, para la visualización e interpretación de la información obtenida mediante la clasificación de contenido en comentarios de twitter.

4.2 Objetivos Específicos:

- Diseñar le estructura de la aplicación web basados en el modelo vista controlador.
- Implementar la parte del controlador, para permitir la comunicación entre vista y modelo.
- Implementar la parte del modelo donde se procesará las peticiones del cliente.
- Implementar la vista, que generara gráfica y muestra resultados.
- Implementar los paquetes de twitter para el uso del API en la aplicación web.
- Aplicar algoritmo de clasificación de datos sobre los comentarios de Twitter.
- Modelar un escenario real con tuits reales en tiempo real.

5. Marco Teórico

5.1 Minería de Datos (Data Mining)

La minería de datos es la práctica de la búsqueda automática en grandes volúmenes de datos para descubrir patrones y tendencias que van más allá de un análisis convencional. La minería de datos utiliza algoritmos matemáticos sofisticados para clasificar los datos y evaluar la probabilidad de eventos futuros.

La minería de datos puede responder a preguntas que no pueden ser abordadas a través de técnicas de simples consultas.

Formas de la minería de datos:

Predicción:

Muchas formas de minería de datos son predictivas. Por ejemplo, un modelo puede predecir el ingreso basado en la educación y otros factores demográficos. Las predicciones tienen una probabilidad asociada (¿Qué tan probable es esta predicción para ser verdad?).

Agrupamiento:

Otras formas de minería de datos identifican agrupaciones naturales en los datos. Por ejemplo, un modelo puede identificar el segmento de la población que tiene un ingreso dentro de un rango especificado.

La minería de datos devuelve información útil y procesable a partir de colecciones de datos.

5.2 Streaming API de Twitter

El API de streaming de twitter permite conectar nuestra aplicación con twitter. Nos da acceso a los tweets globales en tiempo real. Para poder conectarnos nos pide las Keys que

son datos personales y secretos para la conexión. Una vez que se conecta se pueden hacer consultas, sobre los tweets y la información de los usuarios, y sobre la información de los posts.

Existen diferentes endpoints para el api de streaming:

Public Streams: contiene toda la información pública que fluye a través de twitter. Es ideal para minería de datos, por lo que será la que usaremos en este proyecto.

User Streams: es el flujo de todos los datos correspondientes a un usuario en particular.

Site Streams: Parecida a la user strams pero para varios usuarios.

Las diferencias entre la versión streaming y la REST

La principal característica es que la versión REST realiza una petición de conexión a twitter por cada una de las solicitudes que el usuario realiza, una vez que se realiza la consulta a twitter esta regresa el resultado de la petición y el servidor la manda al cliente, cerrando la conexión con twitter, como se muestra en la Figura 1.

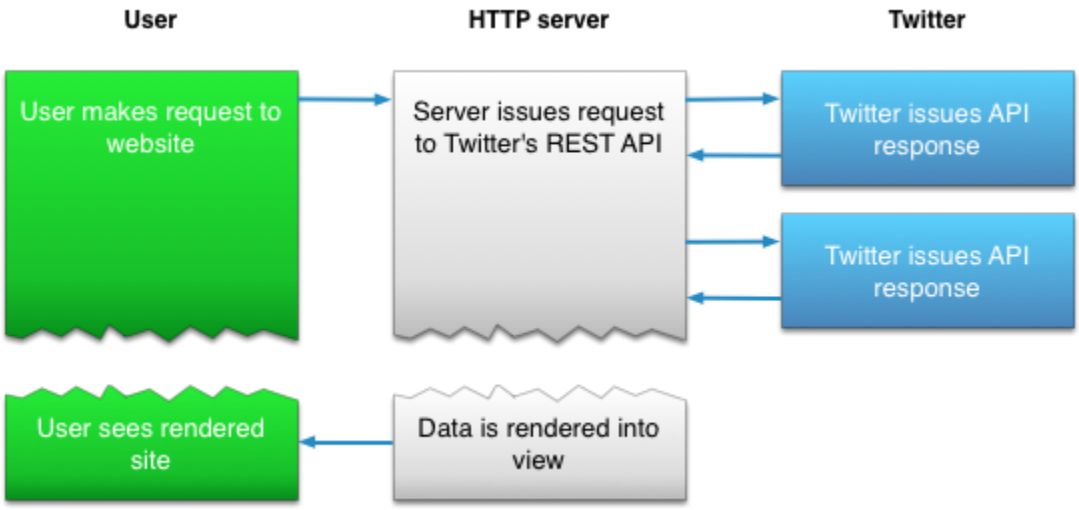


Figura 1. Ejemplo de uso del API REST

Algunos inconvenientes del API REST es que no solo nos mostrara a lo máximo 3200 tuits en total por cada key usada en la aplicación. Y que cada cliente en diferente ventana podrá hacer a lo máximo 15 peticiones por ventana. Si no se respetan estos parámetros twitter podría bloquear las direcciones IP.

Por lo que contando con esas limitaciones se optó que lo ideal para el desarrollo de este proyecto se usara el API Streaming, el cual funciona de la siguiente manera a diferencia del API REST. El API Streaming también se conecta a twitter mediante sus credenciales y keys, solo que este proceso solo se realiza una vez. Ya que se ha establecido conexión con twitter, twitter empieza a compartir el flujo de tuits conforme van ocurriendo en tiempo real, el servidor está recibiendo este flujo y es el servidor el que se encarga de hacer filtros dependiendo las consultas. De tal manera que la conexión no se cierra en cada consulta

esta puede permanecer abierta, y seguir recibiendo los nuevos tuits, como se muestra en la Figura 2. Y es el servidor el que solo manda la información necesaria al cliente, solo aquella que es la que solicito.

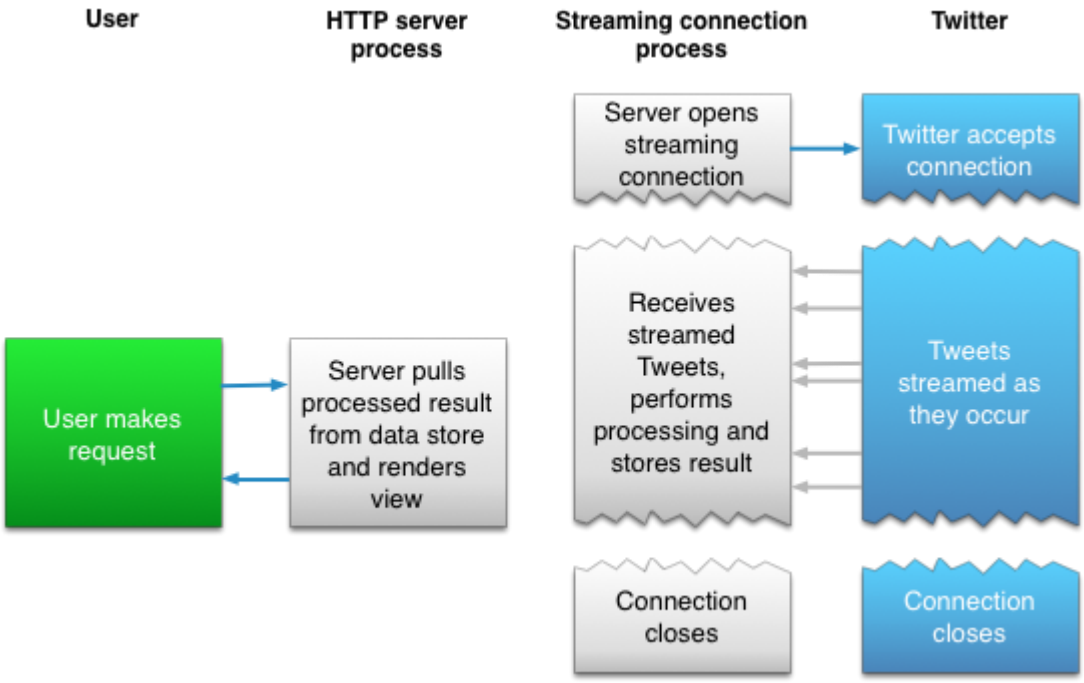


Figura 2. Ejemplo del API Streming

La conexión Streaming con twitter es ideal para muchas aplicaciones, ya que no hay límite de conexiones, ni de tweets analizados, solo que no muestra todos los tweets, sino solo algunos de ellos. También repite muchos tweets, y estos podrían causar basura en nuestra aplicación. La documentación de Twitter no dice que porcentaje de tuits en tiempo real muestra ni con cuanto tiempo de retraso. Si uno quiere realmente tener acceso a la mayor cantidad de tuits en tiempo real para su aplicación, es necesario usar el API REAL-TIME vía gnip que es de comercial. En este proyecto se utilizó el API Streaming de Twitter.

5.3 Formato de un Tweet

Los tweets son estructuras de objetos que contienen datos tanto del propio tuit, como fecha de creación, id, hashtag, contador, hora, geo localización, etc. Así como datos del usuario, como nombre, id, foto, seguidores, amigos, lenguaje, localización, etc. Todos estos datos

estén incluidos en este arreglo de objetos, que son transmitidos en formato JSON. A continuación, mostraremos un ejemplo de cómo es la estructura de un twitter:

{ created_at: 'Tue Mar 21 01:24:10 +0000 2017',
  id: 843996623072124900,
  id_str: '843996623072124928',
  text: 'Tiene México el "cementerio" clandestino más grande de AL https://t.co/rzLsUAl4bM',
  source: 'Twitter for Android',
  truncated: false,
  in_reply_to_status_id: null,
  in_reply_to_status_id_str: null,
  in_reply_to_user_id: null,
  in_reply_to_user_id_str: null,
  in_reply_to_screen_name: null,
  user:
   { id: 1096406485,
     id_str: '1096406485',
     name: 'jorge nava alvarado',
     screen_name: 'NavaJorgenava',
     location: 'mexico distrito federal',
     url: null,
     description: 'consultoria estrategica, analista en temas de gobierno, proyectista de politicas publicas, emprendedor en ideas referidas a temas de gobierno.',
     protected: false,
     verified: false,
     followers_count: 981,
     friends_count: 4990,
     listed_count: 2,
     favourites_count: 32,
     statuses_count: 3089,
     created_at: 'Wed Jan 16 21:25:24 +0000 2013',
     utc_offset: -18000,
     time_zone: 'Central Time (US & Canada)',
     geo_enabled: false,
     lang: 'es',
     contributors_enabled: false,
     is_translator: false,
     profile_background_color: 'C0DEED',
     profile_background_image_url: 'http://abs.twimg.com/images/themes/theme1/bg.png',
     profile_background_image_url_https: 'https://abs.twimg.com/images/themes/theme1/bg.png',
     profile_background_tile: true,
     profile_link_color: '0084B4',
     profile_sidebar_border_color: 'FFFFFF',
     profile_sidebar_fill_color: 'DDEEF6',
     profile_text_color: '333333',
     profile_use_background_image: true,

```
    profile_image_url:
'http://pbs.twimg.com/profile_images/657260513164816384/RxjaubKf_normal.jpg',
    profile_image_url_https:
'https://pbs.twimg.com/profile_images/657260513164816384/RxjaubKf_normal.jpg',
    default_profile: false,
    default_profile_image: false,
    following: null,
    follow_request_sent: null,
    notifications: null },
  geo: null,
  coordinates: null,
  place: null,
  contributors: null,
  is_quote_status: false,
  retweet_count: 0,
  favorite_count: 0,
  entities:
   { hashtags: [],
     urls: [ [Object] ],
     user_mentions: [],
     symbols: [] },
  favorited: false,
  retweeted: false,
  possibly_sensitive: false,
  filter_level: 'low',
  lang: 'es',
  timestamp_ms: '1490059450296' }
```


Prácticamente se puede hacer filtros para cada objeto del twittee, y así hacer consultas más específicas, por ejemplo en este proyecto filtraremos todos los tuits que son de “lang: ‘es’ ” esto quiere decir que solo analizaremos los tuits que estén en idioma español aunque podríamos hacer otros filtros como por ‘time_zone’ o por ‘location’. O prácticamente cualquier otro campo contenido en nuestra estructura del Twitter.

5.4 Lista AFINN

AFINN es una lista de palabras en inglés clasificada por enteremos con valores entre menos cinco (muy negativo) y más cinco (muy positivo). Estas palabras fueron etiquetadas de manera manual por Finn Arup Nielsen en 2009-2011. Contiene 2477 palabras y esta lista es empleada para análisis de sentimientos, la minería de opinión, y minería de texto. Para este proyecto con la ayuda de la Doc. Ángeles Belén Priego Sánchez del departamento de sistemas en la Universidad Autónoma Metropolitana Azcapotzalco, se tradujo la lista de palabras AFFIN al español y se agregaron algunas nuevas palabras para una mejor clasificación de los comentarios de twitter, como se muestra en la Figura 3.

```
rechazar   -1
Rechazado  -2
rescate    2
rescatada  2
Rescata    2
resentido  -2
Renunciar  -1
resignado  -1
Renunciando      -1
Renuncia   -1
resuelto   2
resolver   2
resuelto   2
Resuelve   2
ResoluciÃ³n      2
respetado  2
responsable      2
sensible   2
sosegado   2
inquieto   -2
restaurar  1
Restaurado 1
Restaura   1
Restaurando      1
restringir -2
restringido      -2
Restringiendo    -2
restricciÃ³n     -2
Restringe  -2
Retenido   -1
retardar   -2
retrasado  -2
retirada   -1
venganza   -2
vengativo  -2
Venerado   2
reanimar   2
Revive     2
recompensa 2
Recompensado     2
gratificante     2
Recompensas      2
Rico 2
ridÃculo   -3
aparejo    -1
Aparejado  -1
direcciÃ³n correcta   3
```

Figura 3. Ejemplo de la lista AFFIN en español

5.5 NodeJS

Es un entorno de ejecución para JS construido con el motor de JavaScript V8 de Chrome. Usa un modelo de operaciones E/E sin bloque orientado a eventos lo cual permite que sea muy liviano y eficiente. Además, cuenta npm que es el ecosistema más grande de libreas de código abierto en el mundo. NodeJS permite usar JS del lado del servidor, es por ello que se vuelve una herramienta muy poderosa a la ora de implementar, desarrollar y producir.

NodeJS está diseñado como un entorno de ejecución de JavaScript orientado a eventos asíncronos, NodeJS está diseñado para construir aplicaciones en red escalables. En las

aplicaciones se pueden manejar muchas conexiones concurrentes. Por cada conexión el callback será ejecutado, sin embargo, si no hay trabajo que hacer NodeJS estará durmiendo.

Esto contrasta con el modelo de concurrencia que usan hilos del Sistema Operativo. Las operaciones de redes basadas en hilos son relativamente ineficientes y son muy difíciles de usar. Con NodeJS uno se olvida sobre el bloqueo del proceso, ya que no existe. Casi ninguna función en NodeJS realiza I/O directamente, así que el proceso nunca se bloquea. Debido a que no hay bloqueo es muy razonable desarrollar sistemas escalables en NodeJS. A pesar que NodeJS no está diseñado con hilos, no significa que no pueda aprovechar los múltiples núcleos del sistema ya que se pueden lanzar procesos hijos con el API child_process.fork() y el modulo clúster con lo cual se podría realizar balanceo de cargas en los múltiples núcleos a través de sockets.

La versión de NodeJS utilizada en esta propuesta fue la versión v7.9.0 la cual se distribuye de manera libre y gratuita bajo la colaboración de Linux Fundatión.

6. Desarrollo del Proyecto

6.1 Obtención de los Keys y Tokens de Twitter

Para este punto se usó una cuenta real de twitter previamente registrada y se accedió a la página de desarrollo de twitter, donde pide información personal para ser parte del equipo de desarrollo de twitter. Después que uno ha sido registrado uno se loguea y accede a la dirección apps.twitter.com donde podemos agregar una nueva app. Para este proyecto se creó la app llamada “tuit_uam” stream tuits, como se muestra en la Figura 4.

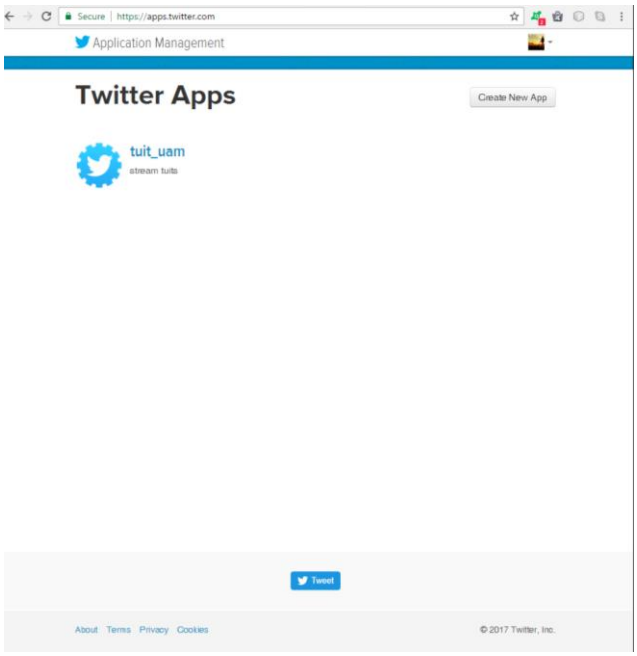


Figura 4. Creación de la aplicación

Una vez que se ha creado la aplicación entonces uno puede acceder a un menú de opciones donde puede, agregar detalles, configuraciones, y lo más importante la obtención de las Keys y Tokens de acceso, como se muestra en la Figura 5. Los cuales nos permitirán acceder a twitter desde nuestra aplicación web, sin ellas no podríamos realizar la conexión a twitter.

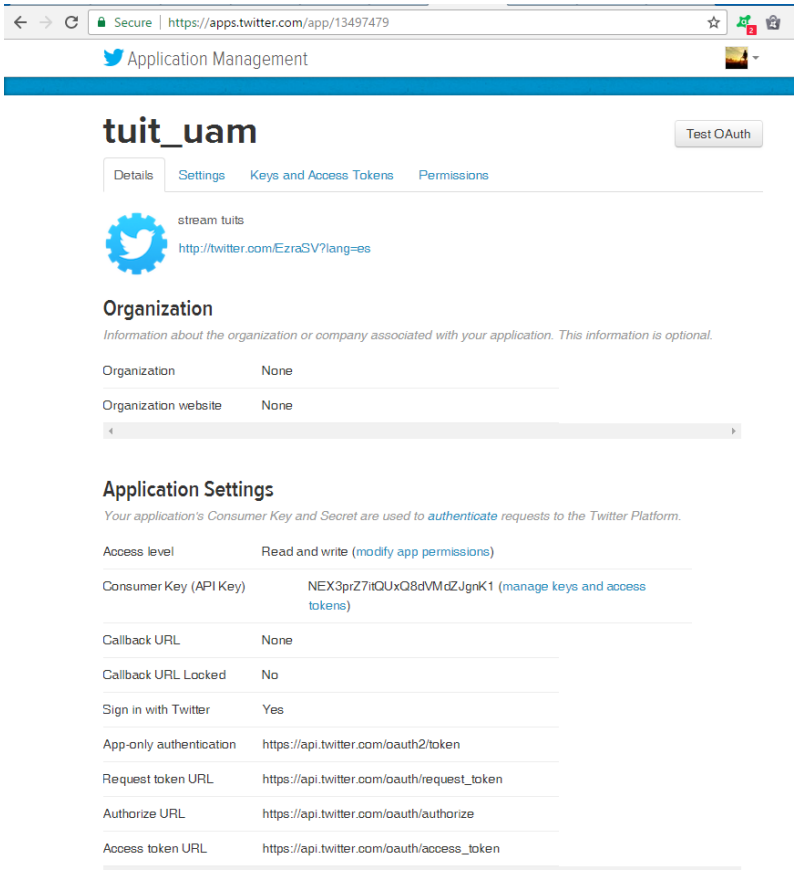


Figura 5. Ventana para solicitud de keys y tokens

Una vez que tenemos nuestras keys y tokens, entonces podremos agregarlas en nuestra aplicación web para poder conectarnos con twitter. Estas keys y tokens tienen fecha de vencimiento, sin embargo, las keys y tokens usados en la aplicación web no necesariamente tienen que ser las relacionadas a un solo usuario de twitter, pueden generarse nuevas keys y tokens en lo futuro por cualquier otro usuario e implementarlas en la aplicación.

6.2 Instalación de NodeJS

NodeJS está disponible para Linux, Windows y Mac, puesto que en este proyecto se utilizó una computadora con Windows se descargó el ejecutable de NojeJS v7.9.0 para Windows

desde la página oficial https://nodejs.org/es/download/ y se instaló siguiendo el proceso de instalación asistido por el mismo ejecutable.

6.3 Instalación de los paquetes npm

Para este proyecto se utilizaron los siguientes paquetes npm:

Ntwitter : Permite la conexión con twitter mediante las keys y tokens y el uso del Streaming API de Twitter para NodeJS.

Sentiment: Usa la lista AFINN modificada para valorar las palabras en aplicaciones NodeJS.

Express: es un rápido y minimalista web framework para NodeJS.

Para instalar cada uno de estos paquetes solo hay que abrir la consola de NodeJS y escribir el comando *intall* seguido del paquete que queremos instalar, como se muestra en la Figura 6.

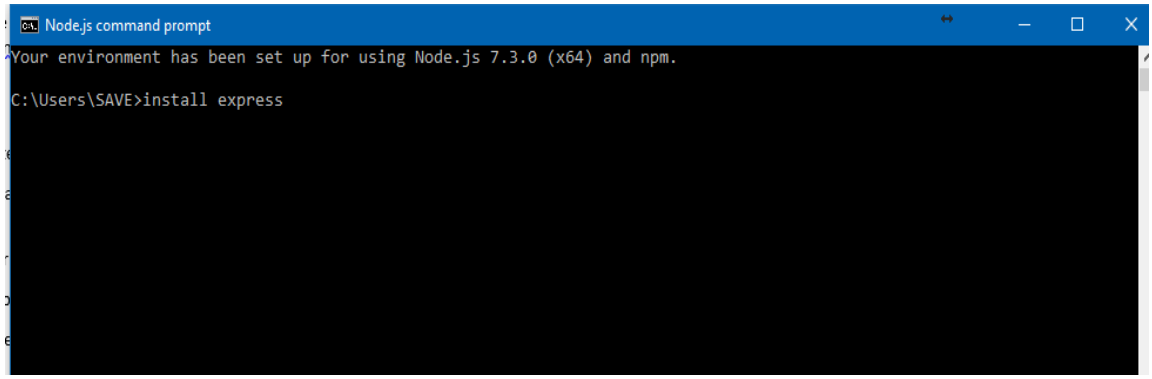


Figura 6. Consola NodeJS con ejemplo de comando install

6.4 Estructura de la Aplicación

La estructura de las aplicaciones en NodeJS es la siguiente, se tiene una carpeta con el nombre de nuestra aplicación dentro de esta carpeta se alojará todos los archivos y recursos de nuestra aplicación, como muestra la Figura 7. Las carpetas importantes y por default al crear cualquier aplicación NodeJS son:

.idea : contiene los archivos XML que mapean, módulos, jslibreria, espacio de trabajo y configuraciones de ejecución. Esta carpeta se genera por default y no hay que hacerle modificaciones.

bin: contiene el archivo www que contiene las configuraciones para el levantamiento del servidor. También esta carpeta se genera por default y no se requiere modificar.

node_modules: Esta carpeta es muy importante ya que contiene todos los paquetes o módulos npm que utilizara nuestra aplicación, incluye tanto los paquetes por default como aquellos que nosotros vayamos a instalar o crear desde cero. Si se elimina esta carpeta o los archivos contenidos en su interior la aplicación dejara de funcionar.

public: Esta carpeta nosotros la creamos manualmente por lo que el nombre puede cambiar, esta carpeta se usara para acceder a contenido dentro de nuestra aplicación web, por ejemplo, contiene los archivos CSS, Imágenes, y código JS para la parte del cliente, también en ella se guardara información de los tweets analizados generando archivo JSON.

routes: Esta carpeta es muy importante ya que contiene las rutas de nuestra aplicación web, es decir contiene los archivos JS con todo el código de procesamiento pesado que realizara la aplicación web. En este proyecto solo hay una ruta que es el index.

views: Contiene las vistas en archivos.hjs que prácticamente es toda la parte visual en html5 y css3, estos archivos hjs son llamados por la aplicación y finalmente mandados al navegador del cliente. Esto permite tener separada la parte vista de la parte operativa. Parecido al modelo vista controlador.

app.js: este archivo es el encargado de levantar el servidor e inicializar la aplicación. Entre otras cosas nos permite asignar los datos de puerto del servidor, indicar que paquetes se requieren, el uso de las rutas, y mantiene al servidor en espera de solicitudes. Es vital y de suma importancia este archivo.

package.json : es un archivo de tipo informativo, nos indica el nombre de la aplicación, que versión es, el autor, los paquetes que utiliza, cual es la ruta de inicio entre otras cosas.

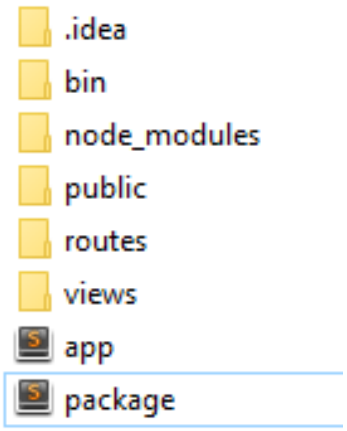


Figura 7. Estructura de la aplicación NodeJS

6.5 Diseño visual de la página Web

En esta parte se diseñó la parte visual de la aplicación usando HTML5 y CSS3. Para el diseño se buscó que pudiera ser claro y minimalista, ya que una página con muchos botones e información llega a ser bastante complicada o difícil de operar. En este caso solo se requiere un campo de entrada en este caso un input box y un botón submit. Esto es para la página de inicio, como muestra la Figura 9. El logo se diseñó pensando en las herramientas convencionales que son una pala y un pico y encerrados en un globo de conversación haciendo alusión a minería de comentarios. El color azul fue utilizado por que es referente al azul de twitter, aunque en un tono más obscuro. La leyenda “sentimientos”

hace referencia a que se analizan sentimientos o se mina el sentimiento del comentario de twitter, como muestra la Figura 8.

Figura 8. Logo para la aplicación web

Figura 9. Vista principal del Index

La siguiente vista es la que muestra los resultados del contenido a analizar, mostrando resultados y la gráfica con la cantidad de comentarios positivos, negativos y neutros. y un botón para realizar otra consulta y la lista de los tuits analizados y ya ordenados dependiendo si son positivos, neutros o negativos, como muestra la Figura 10.

Figura 10. Vista de resultados

6.6 Procesamiento de los tweets

En esta etapa se van capturando todos los tweets que están en el flujo de datos de twitter, pero como hemos limitado la aplicación a solo contenido en español descarta los demás tweets y solo nos muestra los que están en español. Después en base a la palabra que hemos seleccionado para analizar empieza a analizar solo los tweets que contengan esa palabra.

Para poder asignarle una polaridad al contenido de los tweets estos se dividen en sus palabras y cada palabra va buscando su valor en la lista AFFIN si esta se encuentra se asigna su valor dependiendo el rango que va desde -5 muy negativo hasta +5 muy positivo, de esta manera se saca un total sumando el valor de sentimiento de cada una de las palabras, aquellas que no tienen una referencia en la lista AFINN estas toman el valor de cero. Y así podemos saber si el comentario es positivo, negativo o neutro. Por ejemplo:

Supongamos que analizaremos el siguiente comentario

“La pizza está muy rica”

Lo primero es tokenizar el texto divido por espacios

La

pizza

esta

muy

rica

Después se le asigna a cada palabra un valor según corresponda en la lista AFINN

La 0

Pizza 0

Esta 0

muy 0

rica 5

sentimiento total = 0+0+0+0+5 = 5

Porcentaje de Sentimiento = sentimiento total / número de tokens con valor diferente a 0

Porcentaje de Sentimiento = 5 / 1 = 5

Para este proyecto se usó un sistema escalado donde el umbral va desde el 5 muy positivo al -5 muy negativo, y por análisis experimentales y que mejor funciono para este proyecto se usó la siguiente formula.

$$sentimiento \begin{cases} \geq 0.5 \; Positivo \\ \leq -0.5 \; Negativo \\ > -0.5 \; y < 0.5 \; Neutro \end{cases}$$

5 > 0.5 por lo tanto este tweet es Positivo

6.7 Delimitando Tweetts a idioma español

Ya que el streaming nos arroja un muestreo de todos los tweets globales, es necesario acotar solo los tweets que nos sirven para analizar y cuáles no. Ya que en este proyecto la lista AFINN contiene palabras en inglés y en español, esto quiere decir que solo podrá procesar y valorar los tweets en inglés y en español. Por lo que los demás idiomas no tendrían caso analizar ya que serían pasados con un valor de cero mandándolos a la lista de los neutros. Eso solo ocasionaría ruido o basura en los resultados.

Por ello acotaremos el análisis solo a los tweets de idioma español, de esta manera solo analizaremos del flujo del streaming todos los tweets que sean de lenguaje español “lang : es ”, como se muestra en la Figura 11. Incluso se descartaron los tweets en inglés, porque se diseñó para ser una herramienta primeramente para el español, sin embargo, si se quisiera analizar otro idioma, solo se tendría que agregar los pesos de las palabras en ese idioma en la lista AFFIN y después realizar la acotación correspondiente al idioma en que se deseen analizar los tweets.

```
    // Solo evaluamos tuits en espanol
if (data.lang === 'es' ) {
```

Figura 11. Delimitando Tweets a idioma español.

También se puede delimitar los tweetts por otros campos no solo el idioma, por ejemplo, se podría delimitar por localización, ubicación, usuarios, región, timezone y prácticamente cualquier otro campo que este incluido en la estructura del tweet. Sin embargo, ya que esos valores no siempre aparecen llenos, y muchas veces aparecen como campos vacíos, al no tener valores en esos campos estos son eliminados en automático. Y si la información y los datos es mínima o incierta, esto solo daría resultados inciertos.

6.8 Presentación de los Resultados

Los resultados obtenidos son los propios tweets, que ya han sido filtrados por idioma, y que han pasado a la parte procesamiento para ser clasificados, ya que su estructura es un arreglo de objetos podemos manipular y manejar esos datos como objetos.

Para poder Gráficar los resultados se inicializan cuatro variables cada vez que se inicializa una nueva consulta, estas variables corresponden al total de tweets analizados, al total de positivos, negativos, y neutros. Estos valores se pasan como parámetros a la petición request de http, y son los que alimentan la vista de resultados con los valores para Gráficar el resultado, como muestra la Figura 12.

Figura 12. Ejemplo de presentación de los tweets analizados

Para la visualización de le los tweets que se están analizando, primero son almacenados en tres archivos diferentes con formato JSON y codificación UTF8, como se muestra en la Figura 13. Y son guardados para más adelante extraer su contenido y visualizarlo en la parte inferior de la aplicación web.

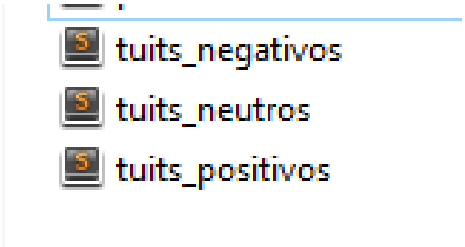


Figura 13. Archivos JSON con los tweets ya clasificados.

Una vez que se ha clasificado a qué tipo de polaridad pertenece el tweet, entonces se mandan a escribir a pantalla, para visualizarlos en la vista de resultados, como muestra la Figura 14.

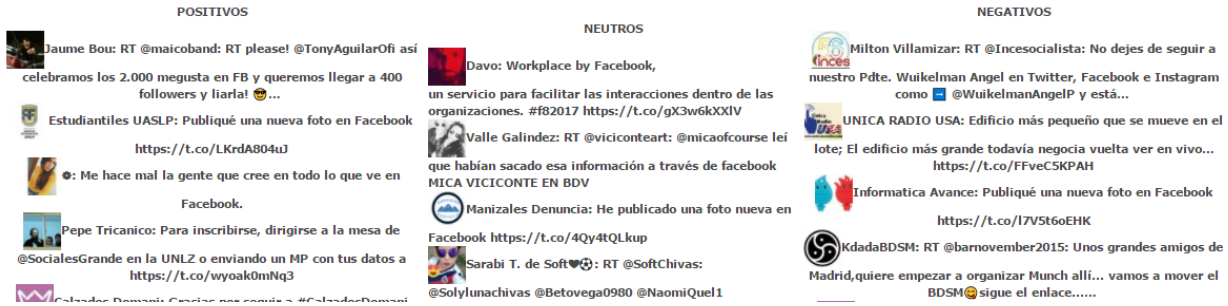


Figura 14. Visualización de los tweets ya clasificados.

Los archivos JSON mantienen la estructura del tweet por lo que es fácil recuperar su información, incluso, uno puede indicar que componentes del tweet desea guardar en el archivo JSON y cuales no por ejemplo para este proyecto solo nos importa el nombre del usuario, fecha de creación, el texto y la foto de perfil. Y todo esto lo guardamos en una variable que será escrita en el archivo JSON, como se muestra en la Figura 15.

```
var tt = {nombre: data.user.name, texto: data.text, foto: data.user.profile_image_url};
```

Figura 15. Variable con los datos del tweett

Esta variable ahora contiene la estructura del tweett que hemos clasificado. El cual agregaremos a un arreglo de objetos dependiendo su clasificación. Por ejemplo, si el tweett es positivo lo agregamos al arreglo positivo, como se muestra en la Figura 16.

```
ttw_p.push(tt);
```

Figura 16. Agregando un nuevo tweet positivo a nuestro arreglo positivo

Finalmente, se escribe en el archivo JSON correspondiente en este caso en el de tweets positivos, como se muestra en la Figura 17.

```
var json = JSON.stringify(ttw_p);
fs.writeFile("public/tuits_positivos.json", json,'utf8', (err) => {
```

Figura 17. Guardando el arreglo de objetos positivos en un archivo JSON para tweets positivos

Esto se hace para todos los tweets que han sido analizados y dependiendo de su polaridad estos son escritos en su respectivo archivo.

7. Pruebas

Para realizar las pruebas se instaló NodeJS en una computadora de escritorio con Windows 10 64bits, procesador Intel Core i5 y 8Gb de Ram. Una vez instalado en entorno NodeJS, se copió la carpeta que contiene todos los archivos de la aplicación a una ruta en el disco principal, específicamente en la carpeta usuarios, como se muestra en la Figura 18.

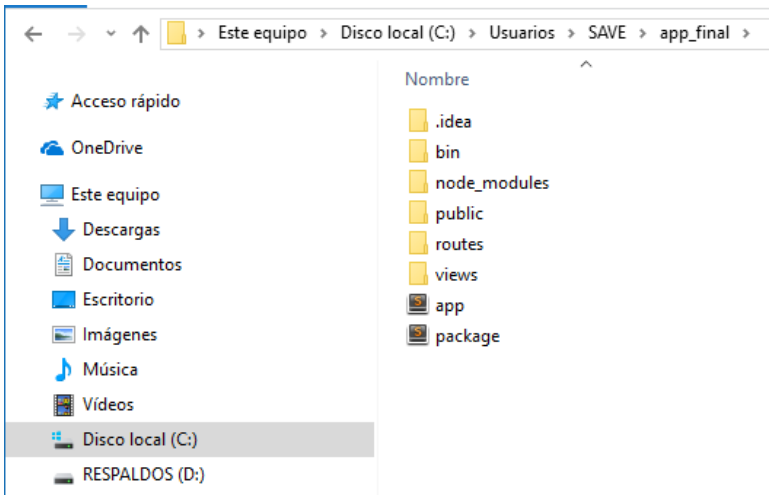


Figura 18. Ruta de la carpeta de la aplicación

Una vez la carpeta está alojada en la ruta que se desea, ahora solo hay que abrir una terminal de NodeJS. En la cual accederemos a la ruta donde hemos guardado nuestra aplicación, como se muestra en la Figura 19.

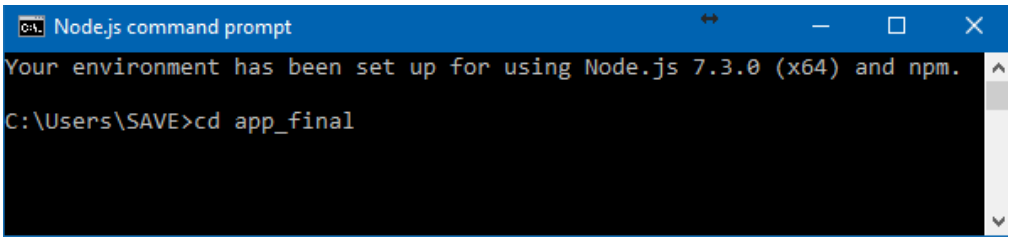


Figura 19. Accediendo a la ruta de nuestra aplicación

Finalmente, para ejecutar la aplicación, una vez que se está dentro de la ruta se escribe el comando *node app.js*, como se muestra en la Figura 20. El archivo app.js es el que se encargara de inicializar el servidor que alojara nuestra aplicación web.

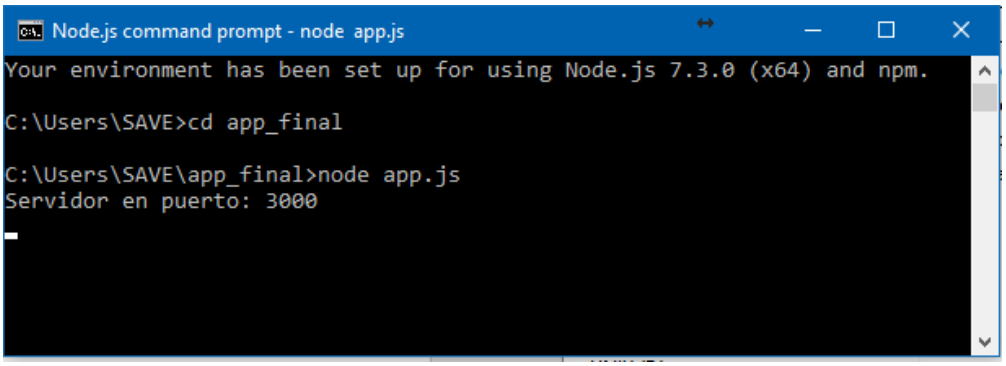


Figura 20. Ejecutando la aplicación en el puerto 3000

Una vez que el servidor se levantó y está corriendo, en este proyecto se usó por default el puerto 3000, ahora solo hay que ir al navegador web e ingresar al localhost:3000, como se muestra en la Figura 21.

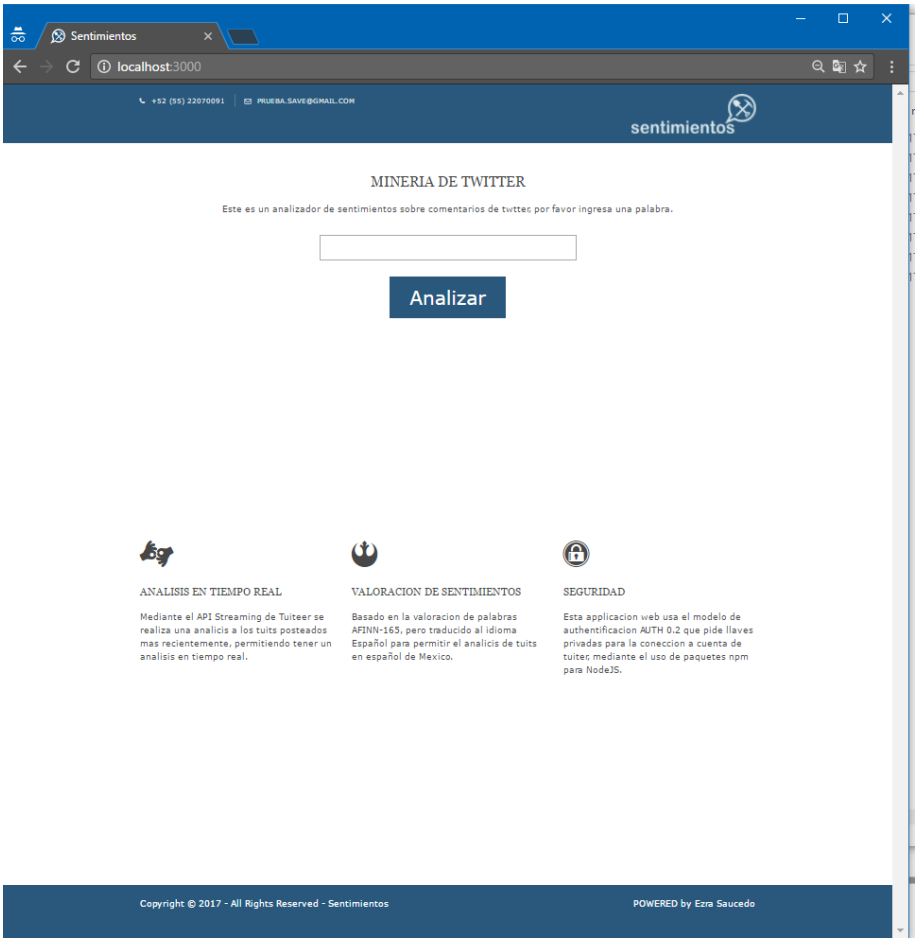


Figura 21. Navegador Web Chrome visualizando la aplicación.

Para esta prueba se realizó una consulta para saber los tweets relacionados a “semana santa”, la consulta se dejó correr por un tiempo de 2 min en lo cual se analizaron 24 tweets

de los cuales 2 fueron positivos, 9 fueron negativos y 13 fueron neutros, como muestra la Figura 22.

Figura 22. Prueba real analizando tweets de "Semana Santa"

En los archivos JSON, podremos encontrar los tweets analizados correspondientemente, como se muestra en la Figura 23.

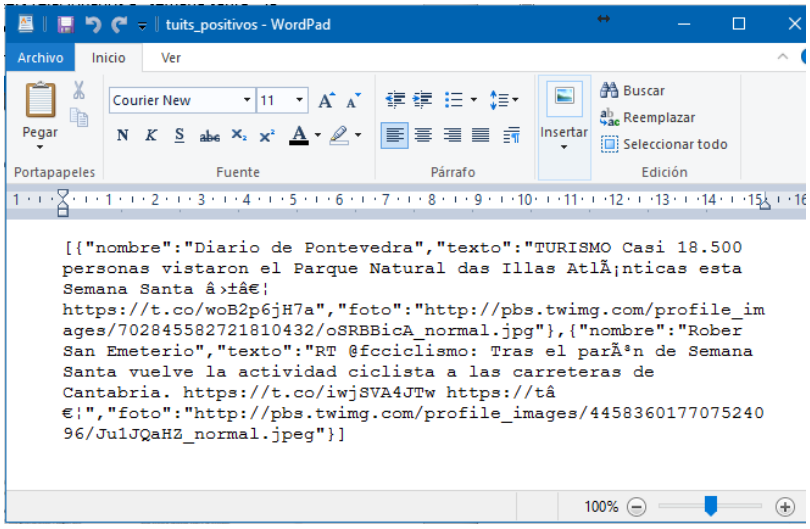


Figura 23. Archivo JSON con los Tweets positivos de "Semana Santa"

Posteriormente se realizaron 3 pruebas más por intervalos de tiempo, para poder ver como cambiaba la polaridad de los tweets con respecto al tiempo. Para esta prueba se analizaron las palabras “ciudad de México”, “metro”, y “Justin Bieber” a lo largo de 20min tomando muestras cada 5 minutos, como se muestra de la Figura 24 a la Figura 35.

Figura 24. Analizando “Ciudad de Mexico” a los 5min

Figura 25. Analizando “Ciudad de Mexico” a los 10min

Figura 26. Analizando "Ciudad de Mexico" a los 15min

Figura 27. Analizando "Ciudad de Mexico" a los 20min

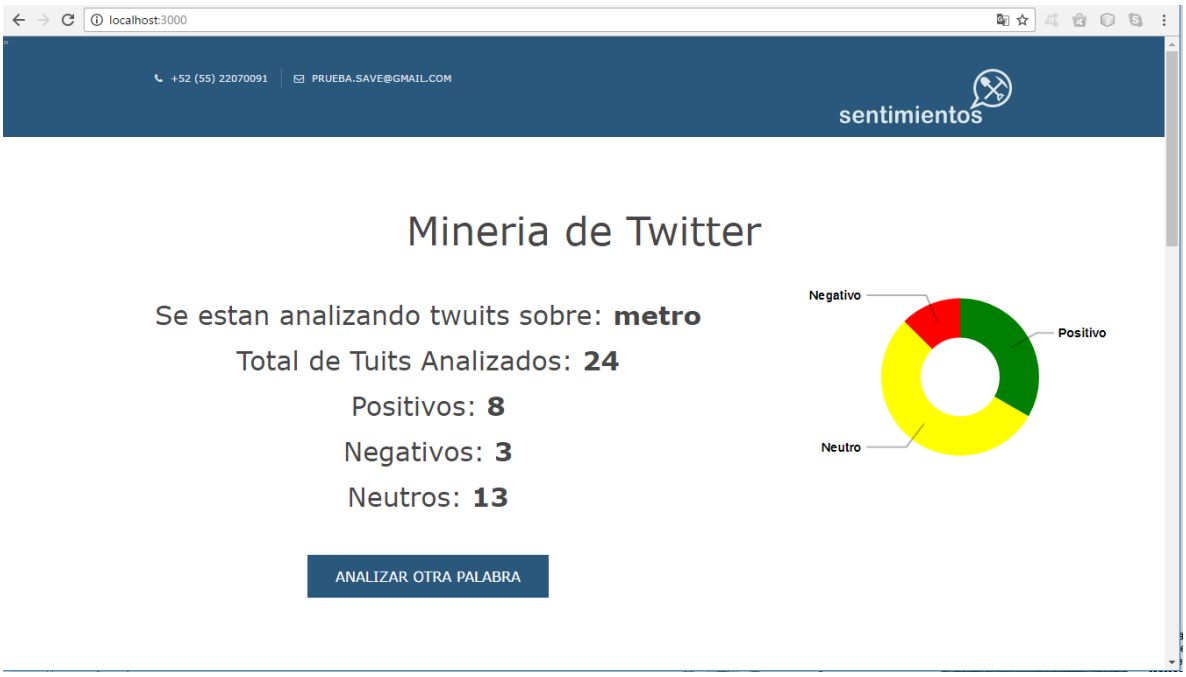


Figura 28. Analizando "metro" a los 5min

Figura 29. Analizando "metro" a los 10min

Figura 30. Analizando "metro" a los 15min

Figura 31. Analizando "metro" a los 20min

Figura 32. Analizando "Justin Bieber" a los 5min

Figura 33. Analizando "Justin Bieber" a los 10min

Figura 34. Analizando "Justin Bieber" a los 15min

Figura 35. Analizando "Justin Bieber" a los 20min

8. Análisis y Discusión de Resultados

Una vez realizadas las pruebas anteriores analizaremos los resultados obtenidos, para ello usaremos una tabla y posteriormente Gráficaremos los datos para mostrar el comportamiento que estos tienen con respecto al tiempo.

Algunas de las variables a considerar son el total de tweets analizados para cada una de las muestras, cabe recalcar que los tópicos que se escogieron fueron de diferentes sectores por ejemplo para las pruebas de realizo una consulta a ciudad, servicio y persona (Ciudad de México, Metro y Justin Bieber).

Primero mostraremos los datos obtenidos en cada consulta de manera general como se muestra en la Tabla 1. Y posteriormente se mostrará la información en tablas independientes para cada una de las consultas como se muestra en las Tablas 2, 3 y 4.

Tópico	Total de Tweets	Tweets Positivos	Tweets Negativos	Tweets Neutros
Ciudad de México	44	0	33	11
Metro	272	11	246	15
Justin Bieber	240	221	9	10

Tabla 1. Resultado de las 3 consultas durante 20min

Ciudad de México				
Tiempo	**Total de Tweets**	**Tweets Positivos**	**Tweets Negativos**	**Tweets Neutros**
5 min	4	0	3	1
10 min	25	0	14	11
15 min	35	0	24	11
20 min	44	0	33	11

Tabla 2. Resultado de la consulta "Ciudad de México"

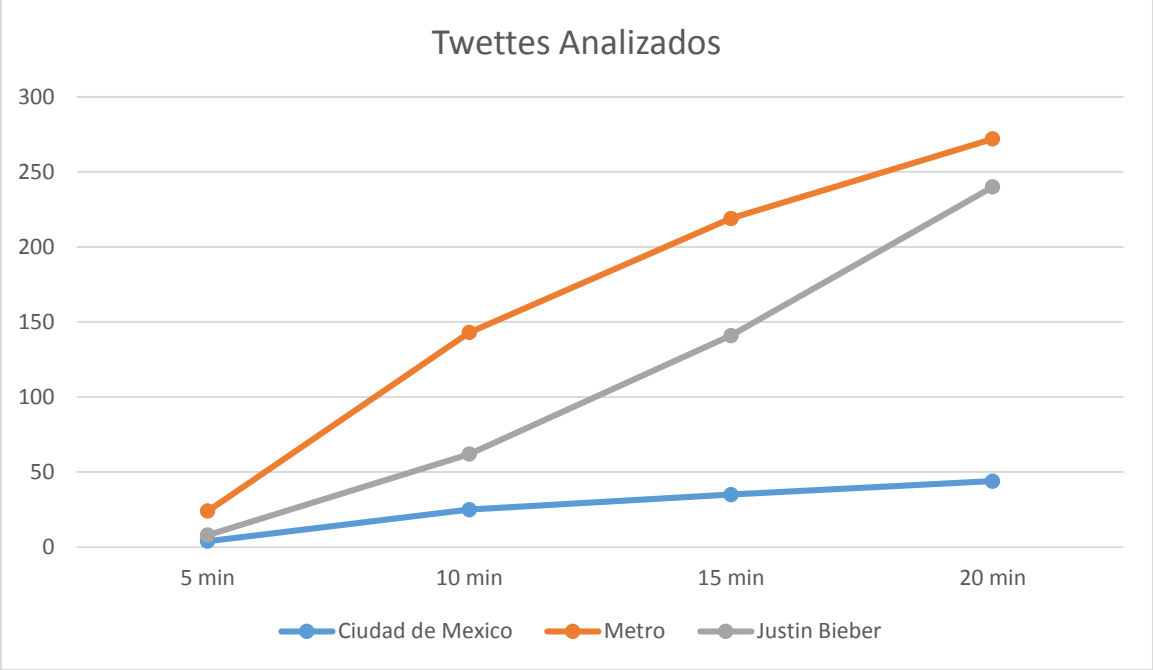
Metro				
Tiempo	**Total de Tweets**	**Tweets Positivos**	**Tweets Negativos**	**Tweets Neutros**
5 min	24	8	3	13
10 min	143	11	117	15
15 min	219	11	193	15
20 min	272	11	246	15

Tabla 3. Resultado de la consulta "Metro"

Justin Bieber				
Tiempo	**Total de Tweets**	**Tweets Positivos**	**Tweets Negativos**	**Tweets Neutros**
5 min	8	6	0	2
10 min	62	43	9	10
15 min	141	122	9	10
20 min	240	221	9	10

Tabla 4. Resultado de la consulta "Justin Bieber"

Finalmente mostramos una gráfica del total de tweets analizados en función del tiempo para poder visualizar como es el comportamiento una de una consulta respecto de la otras en el mismo periodo de tiempo, como se muestra en la Gráfica 1.



Gráfica 1. Total, de tweets analizados en función del tiempo

De esta Gráfica podemos discutir y analizar 2 cosas. Primero que las comparativas fueron muy aleatorias, porque no es posible comparar por ejemplo una ciudad o servicio con una persona, esta gráfica nos indica que contenido o tópico es más mencionado en Twitter en los últimos 20 min y hacia adelante. En estas pruebas podemos ver que Metro fue la que tuvo mayor número de menciones, con respecto a Ciudad de México y a Justin Bieber. Sin embargo, el tiempo realmente de 20 min es un tiempo muy corto, para poder afirmar que su tendencia seguirá al alza, a comparación de los otros tópicos.

9. Conclusiones

La minería en comentarios de twitter, es una herramienta muy útil al momento de querer saber cuál es la percepción o el sentimiento que tienen las personas por algún objeto, servicio, marca, producto, servicio o persona. Sobre todo, si el análisis es sobre comentarios en tiempo real como se plantea en este proyecto, no se analiza información pasada sino lo más reciente, dando una idea más precisa de lo que los usuarios comentan al respecto.

En este proyecto se analizó un solo el sentimiento, de los mensajes buscando clasificarlos dentro de un grupo ya fuese positivo, negativo o neutro. Sin embargo, la polaridad no es lo único que se podría descubrir, también se podría entrenar el algoritmo para identificar y clasificar los comentarios por género, por edad, por nivel educativo, o por nivel económico. Y aprovechar más los datos implícitos del formato de cada tweet para incluso poder mostrar los resultados por tipos de dispositivos en que se twitteo, las zonas que más twittean, o hacer una mezcla de consultas para obtener resultados más claros, e incluso poder no solo Gráficarlos si no poder mostrar mapas, estadísticas, y otras formas de implementaciones que pudieran facilitar la interpretación de los datos.

A partir de los resultados obtenidos se puede decir que el proyecto cumple su finalidad, y valora con un 70% - 80% de correctitud los comentarios de twitter lo cual es bastante bueno. Sin embargo, puede mejorarse y ampliarse la lista AFINN agregando expresiones del lenguaje natural que son utilizadas cada vez con más frecuencia en las redes sociales, o incluso agregar aquellas frases que son de uso específico en ciertas localidades o regiones del país, para poder aumentar la exactitud de la valoración de las consultas. También se podría las raíces de las palabras de tal manera que el algoritmo no busque la palabra si no la raíz de la misma, ampliando así la lista de palabras valoradas permitiendo al algoritmo evaluar y pesar de una manera más correcta cada comentario.

En cuestión de diseño, cabe resaltar que el límite es la imaginación, no hay un parámetro, pero si se trató de respetar una estructura clásica de una aplicación web, los datos analizados pueden ser representados como mejor les convengan, en este proyecto se usaron etiquetas con números y una gráfica, con lo cual muestra de manera clara el resultado de las consultas.

Por ultimo cabe recalcar que esta herramienta puede ser implementada para dispositivos móviles, o como parte de otras aplicaciones web, el análisis de los datos mediante técnicas de minería de datos es cada vez una herramienta más poderosa y necesaria para las empresas, marcas, sectores públicos, industrias y gobiernos. Permite corregir problemas lo antes posible, cambiar estrategias de mercado, saber las tendencias y las aceptaciones del público, entre otros.

La minería en comentarios tanto de Twitter, así como de otras redes sociales seguirá siendo un campo de estudio cada vez más y mejor explotado.

10. Bibliografía

[1]”Sistema visualizador de información extraída de modelos ontológicos del dominio académico basado en web", Licenciatura, Universidad Autónoma Metropolitana, 2016.
[2]”Aplicación web de administración de horarios para estudiantes", Licenciatura, Universidad Autónoma Metropolitana, 2009.
[3]”ESTANDARES EN EL DISEÑO DE LOS SITIOS WEB DE INSTITUCIONES EDUCATIVAS DE NIVEL SUPERIOR", Maestría, Universidad Autónoma Metropolitana, 2008.
[4]”Herramienta para el preprocesamiento de tweets con base en búsqueda por tópico", Maestría, Centro de Investigación y De Estudios Avanzados Del Instituto Politécnico Nacional, 2015.
[5]”SemanticWebBuilder", *SemanticWebBuilder*, 2016. [Online]. Available: http://www.semanticwebbuilder.org.mx/swb/swb/SWBSocial. [Accessed: 10- Nov- 2016].
[6]”Hashtag Tracking for Twitter, Instagram and Facebook - Keyhole", *Keyhole.co*, 2016. [Online]. Available: http://keyhole.co/preview. [Accessed: 10- Nov- 2016].

11. Anexos

11.1 Archivo app.js

```
var port = (process.env.VCAP_APP_PORT || 3000);

var express = require('express');
var path = require('path');
var favicon = require('serve-favicon');
var logger = require('morgan');
var cookieParser = require('cookie-parser');
var bodyParser = require('body-parser');
var sentiment = require('sentiment');
var twitter = require('ntwitter');
var index = require('./routes/index');
var users = require('./routes/users');
var app = express();
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'hjs');
app.use(logger('dev'));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));
app.use('/', index);
app.use('/users', users);

// error 404
app.use(function(req, res, next) {
  var err = new Error('Not Found');
  err.status = 404;
  next(err);
});

// error
app.use(function(err, req, res, next) {

  // set locals
  res.locals.message = err.message;
  res.locals.error = req.app.get('env') === 'development' ? err : {};

  // render pagina de error
  res.status(err.status || 500);
  res.render('error');
});

module.exports = app;

app.listen(port);
console.log("Servidor en puerto: " + port);
```

11.2 Archivo Index.js

```
var express = require('express');
var router = express.Router();
var sentiment = require('sentiment');
var twitter = require('ntwitter');
var fs = require('fs');

// llaves para coneccion a tweteer
var tweeter = new twitter({
    consumer_key: 'NEX3prZ7itQUxQ8dVMdZJgnK1',
    consumer_secret:
'8hw2KmMKUld05ATSZn3oeDNsS84LjRMznyHO9SaV0k1F7oKNkV',
    access_token_key: '1287171786-
O5SCayV6FwToFCuDwEY2KrAkLNR507OH59STXJL',
    access_token_secret:
'VqtPXjPvfwVRQezCD3ch2IIKJUStTAW6ankAzQqRxk6Yl'
});

var contadorTuits = 0;
var sentimientoTotal = 0;
var positivo = 0;
var negativo = 0;
var neutro = 0;
var fraseaAnalizar = null ;
var ttw_p=[];
var ttw_neg=[];
var ttw_neu=[];

function reiniciarAnalisis() {
    // if (stream) {
    // var tempStream = stream;

    /// stream = null;
    //tempStream.destroy();
    //  stream.on('end', function (response) {
    // Handle a disconnection
    //});
    // }
    fraseaAnalizar = null ;

}

function iniciarAnalisis(frase) {
    var stream;
    // limpia si volvemos a iniciar el monitoreo
    if (fraseaAnalizar) {
        reiniciarAnalisis();
    }
    fraseaAnalizar = frase;
```

```
contadorTuits = 0;
sentimientoTotal = 0;
positivo = 0;
negativo = 0;
neutro = 0;
ttw_p=[];
ttw_neg=[];
ttw_neu=[];



tweeter.verifyCredentials(function (error, data) {
    if (error) {
        return "Error al intentar COncctar con Twitter: " + error;
    } else {
        stream = tweeter.stream('statuses/filter', {
            'track': fraseaAnalizar
        }, function (stream) {
            console.log("Monitoring Twitter for " + fraseaAnalizar);
            stream.on('data', function (data) {
                //Mandamos imprimir el tuit en la consola



                // Solo evaluamos tuits en espanol
              if (data.lang === 'es' ) {



            console.log("Tweet #" + contadorTuits +" :"+data.user.name +":  " +
data.user.profile_image_url);
                  //  console.log(data);

                   var tt = {nombre: data.user.name, texto: data.text, foto:
data.user.profile_image_url};



                   // console.log(ttw[0].nombre);
                   sentiment(data.text, function (err, result) {
                       contadorTuits++;
                       sentimientoTotal += result.score;

                       if (sentimientoTotal > 0.5) {
                           positivo++;

                            ttw_p.push(tt);

                            var json = JSON.stringify(ttw_p);
```

```
                        fs.writeFile("public/tuits_positivos.json", json,'utf8', (err) => {
                        if (err) {
                            console.error(err);
                           return;
                            };
                            console.log("Tuit Agregado...");
                            });

                    }
                    else if (sentimientoTotal < -0.5) {
                       negativo++;

                         ttw_neg.push(tt);
                        var json = JSON.stringify(ttw_neg);
                        fs.writeFile("public/tuits_negativos.json", json,'utf8', (err) => {
                        if (err) {
                            console.error(err);
                           return;
                            };
                            console.log("Tuit Agregado...");
                            });
                    }
                    else {
                       neutro++;
                       ttw_neu.push(tt);
                        var json = JSON.stringify(ttw_neu);
                        fs.writeFile("public/tuits_neutros.json", json,'utf8', (err) => {
                        if (err) {
                            console.error(err);
                           return;
                            };
                            console.log("Tuit Agregado...");
                            });
                    }



                });
            }
        });
    });
    return stream;
  }
});
}

function imagenConSentimiento() {
   var avg = sentimientoTotal / contadorTuits;
   if (avg > 0.5) { // Positivo
      return "/images/positivo.png";
   }
```

```
    else if (avg < -0.5) { // Negativo
        return "/images/negativo.png";
    }
    // Neutral
    else { return "/images/neutro.png";}
}


//router.get('/', function (req, res) {
    router.get('/', function(req, res) {

        if (fraseaAnalizar == null) {
    res.render('index');
}
else {


    res.render('analizar', {frase: fraseaAnalizar, total: contadorTuits, pos: positivo,
neg: negativo, neu: neutro });
}

    });


router.get('/analizar', function (req, res) {
    iniciarAnalisis(req.query.frase);
    res.redirect(302, '/');
    //res.render('analizar', { frase: fraseaAnalizar,total:
contadorTuits,pos:positivo,neg:negativo,neu:neutro});
});

router.get('/reiniciar', function (req, res) {
    reiniciarAnalisis();
    res.redirect(302, '/');
});

////////////////////////////////////

function suma(a,b) {


    var sum = a+b;

    return(sum);
}
module.exports = router;
```

11.3 Archivo index.hjs

```
<!DOCTYPE html>

<html lang="">
<head><link rel="icon" type="image/png" href="img/favicon.png" />
<title>Sentimientos</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-
scale=1.0, user-scalable=no">
<link href="css/styles/layout.css" rel="stylesheet" type="text/css" media="all">

<style>


.button {
    background-color: #29587C;
    border: none;
    color: white;
    padding: 15px 32px;
    text-align: center;
    text-decoration: none;
    display: inline-block;
    font-size: 16px;
    margin: 4px 2px;
    cursor: pointer;
}
</style>
</head>

<body id="top">


<div class="wrapper row0">
  <div id="topbar" class="hoc clear" >

    <div class="fl_left">
      <ul>
        <li><i class="fa fa-phone"></i> +52 (55) 22070091</li>
        <li><i class="fa fa-envelope-o"></i> prueba.save@gmail.com</li>
      </ul>
    </div>
    <div class="fl_right">
      <ul>

         <a href="#"> <img src="img/logo.png"></i></a>

     </ul>    </div>     </div></div>

<div class="wrapper row3">
```

```
  <header id="header" class="hoc clear">

   <center><h1>Mineria de Twitter</h1></center>

   <center><article>

         <p class="btmspace-30">Este es un analizador de sentimientos sobre comentarios de twtter, por favor ingresa una palabra.</p>


  <form action="analizar" method= "get">

  <input type="text" name="frase" id="frase" value="" required="required" style=" font-size: 30px;">
  <br>
  <input type="submit" class="button" value="Analizar" style="font-size: 30px;" >

</form>
       </article>
    </center>
  </header>
</div>
 <div class="wrapper row3">
  <main class="hoc container clear">
      <ul class="nospace group">
       </ul>

    <div class="clear"></div>
  </main>
</div>

<div class="wrapper row3">
  <main class="hoc container clear">

    <ul class="nospace group">
      <li class="one_third first">
        <article><i class="btmspace-30 fa fa-3x fa-american-sign-language-interpreting"></i>
          <h6 class="heading font-x1">Analisis en Tiempo Real</h6>
          <p class="btmspace-30">Mediante el API Streaming de Tuiteer se realiza una analicis a los tuits posteados mas recientemente, permitiendo tener un analisis en tiempo real.</p>

        </article>
      </li>
      <li class="one_third">
        <article><i class="btmspace-30 fa fa-3x fa-resistance"></i>
          <h6 class="heading font-x1">Valoracion de Sentimientos</h6>
          <p class="btmspace-30">Basado en la valoracion de palabras AFINN-165, pero traducido al idioma Español para permitir el analicis de tuits en español de Mexico.</p>

        </article>
```

```
      </li>
      <li class="one_third">
        <article><i class="btmspace-30 fa fa-3x fa-expeditedssl"></i>
          <h6 class="heading font-x1">Seguridad</h6>
          <p class="btmspace-30">Esta applicacion web usa el modelo de authentificacion
AUTH 0.2 que pide llaves privadas para la coneccion a cuenta de tuiter, mediante el uso
de paquetes npm para NodeJS.</p>

        </article>
      </li>
    </ul>

    <div class="clear"></div>
  </main>
</div>
<div class="wrapper row3">

  <section class="hoc container clear">

  </section>
</div>

</div>

<div class="wrapper row5">
  <div id="copyright" class="hoc clear">

    <p class="fl_left">Copyright &copy; 2017 - All Rights Reserved - <a
href="#">Sentimientos</a></p>
    <p class="fl_right">POWERED by <a target="_blank" title="Equipo #1">Ezra
Saucedo</a></p>

</div>

<a id="backtotop" href="#top"><i class="fa fa-chevron-up"></i></a>

<script src="/css/scripts/jquery.min.js"></script>
<script src="/css/scripts/jquery.backtotop.js"></script>
<script src="/css/scripts/jquery.mobilemenu.js"></script>



</body>
</html>
```

11.4 Archivo analizar.hjs

```
<!DOCTYPE html>

<html lang="">
<head><link rel="icon" type="image/png" href="img/favicon.png" />
<title>Sentimientos</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-
scale=1.0, user-scalable=no">
<META http-equiv="refresh" content="20"; URL=http://"req.headers.host"/>"
<link href="css/styles/layout.css" rel="stylesheet" type="text/css" media="all">

<style>

#todo {
   width:160%;

}

  #izquierda, #derecha {
      width:35%;
      text-align: center;
}
  #contenido {
      width:35%;
      text-align: center;

}
#izquierda {
   float: left;
}
#derecha {
   float: right;
}
#contenido {
   margin-left: 160px;
}

#div1{float: left;}
#div2{float: right;}

figure {
  font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;
  width: 400%;
  height: 400px;
  position: relative;
}

svg {
```

```
  width: 100%;
  height: 100%;
}

path.slice{
  stroke-width:2px;
}

polyline{
  opacity: .3;
  stroke: black;
  stroke-width: 2px;
  fill: none;
}

.button {
    background-color: #29587C;
    border: none;
    color: white;
    padding: 15px 32px;
    text-align: center;
    text-decoration: none;
    display: inline-block;
    font-size: 16px;
    margin: 4px 2px;
    cursor: pointer;
}
</style>
 <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
 <script src="https://code.jquery.com/jquery-1.10.2.js"></script>
</head>



<body id="top">

<div class="wrapper row0">
  <div id="topbar" class="hoc clear" >

    <div class="fl_left">
      <ul>
        <li><i class="fa fa-phone"></i> +52 (55) 22070091</li>
        <li><i class="fa fa-envelope-o"></i> prueba.save@gmail.com</li>
      </ul>
    </div>
    <div class="fl_right">
      <ul>

         <a href="#"> <img src="img/logo.png"></i></a>
```

```
      </ul>
    </div>

  </div>
</div>

<div class="wrapper row3">
  <header id="header" class="hoc clear">


   <center><p style="font-size: 45px;">Mineria de Twitter</p></center>



   <center>
<article>

<div id="div1">

         <p style="font-size: 30px;" >Se estan analizando twuits sobre:<b> {{frase}} </b></p>

 <p style="font-size: 30px;">Total de Tuits Analizados:<b> {{total}} </b></p>
  <p style="font-size: 30px;" >Positivos:<b> {{pos}}</b></p>
   <p style="font-size: 30px;">Negativos:<b> {{neg}}</b></p>
    <p style="font-size: 30px;">Neutros:<b> {{neu}}<b></p>



  <br>

  <button class="button" onclick="location.href='reiniciar'">ANALIZAR OTRA
PALABRA</button>



</div>

<div id="div2">

<figure>
  <script src="http://d3js.org/d3.v3.min.js"></script>
  <script>
  var x2 = 0 ;
          var svg = d3.select("figure")
      .append("svg")
      .append("g")

  svg.append("g")
      .attr("class", "slices");
  svg.append("g")
```

```
    .attr("class", "labels");
svg.append("g")
    .attr("class", "lines");

var width = 480,
    height = 225,
    radius = Math.min(width, height) / 2;

var pie = d3.layout.pie()
    .sort(null)
    .value(function(d) {
        return d.value;
    });

var arc = d3.svg.arc()
    .outerRadius(radius * 0.8)
    .innerRadius(radius * 0.4);

var outerArc = d3.svg.arc()
    .innerRadius(radius * 0.9)
    .outerRadius(radius * 0.9);

var porcentaje = 0;

svg.attr("transform", "translate(" + width / 2 + "," + height / 2 + ")");

var key = function(d){ return d.data.value; };


var color = d3.scale.ordinal()
    .domain(["Positivo", "Neutro", "Negativo"])
    .range(["green", "yellow", "red"]);



function randomData (){


    var labels = color.domain();
    return labels.map(function(label){

          if(x2==0){
                    x2++;
        return { label: label, value: "{{pos}}" }
    }
        else if(x2==1){
                    x2++;
        return { label: label, value: "{{neu}}" }

    }
```

```
    else{
         x2++;
         return { label: label, value: "{{neg}}" }
    }



    });



}



change(randomData());

d3.select(".randomize")
    .on("click", function(){
        change(randomData());
    });


function change(data) {

    /* ------- PIE SLICES -------*/
    var slice = svg.select(".slices").selectAll("path.slice")
        .data(pie(data), key);

    slice.enter()
        .insert("path")
        .style("fill", function(d) { return color(d.data.label); })
        .attr("class", "slice");

    slice
        .transition().duration(1000)
        .attrTween("d", function(d) {
            this._current = this._current || d;
            var interpolate = d3.interpolate(this._current, d);
            this._current = interpolate(0);
            return function(t) {
                return arc(interpolate(t));
            };
        })

    slice.exit()
        .remove();

    /* ------- TEXT LABELS -------*/

    var text = svg.select(".labels").selectAll("text")
```

```
    .data(pie(data), key);

text.enter()
    .append("text")
    .attr("dy", ".35em")
    .text(function(d) {
      // return d.data.label;

       //porcentaje = porcentaje + d.data.value ;


      return d.data.label;

       //return d.data.value*100;
    });

function midAngle(d){
    return d.startAngle + (d.endAngle - d.startAngle)/2;
}

text.transition().duration(1000)
    .attrTween("transform", function(d) {
        this._current = this._current || d;
        var interpolate = d3.interpolate(this._current, d);
        this._current = interpolate(0);
        return function(t) {
            var d2 = interpolate(t);
            var pos = outerArc.centroid(d2);
            pos[0] = radius * (midAngle(d2) < Math.PI ? 1 : -1);
            return "translate("+ pos +")";
        };
    })
    .styleTween("text-anchor", function(d){
        this._current = this._current || d;
        var interpolate = d3.interpolate(this._current, d);
        this._current = interpolate(0);
        return function(t) {
            var d2 = interpolate(t);
            return midAngle(d2) < Math.PI ? "start":"end";
        };
    });

text.exit()
    .remove();

/* ------- SLICE TO TEXT POLYLINES -------*/

var polyline = svg.select(".lines").selectAll("polyline")
    .data(pie(data), key);

polyline.enter()
```

```
        .append("polyline");

    polyline.transition().duration(1000)
        .attrTween("points", function(d){
            this._current = this._current || d;
            var interpolate = d3.interpolate(this._current, d);
            this._current = interpolate(0);
            return function(t) {
                var d2 = interpolate(t);
                var pos = outerArc.centroid(d2);
                pos[0] = radius * 0.95 * (midAngle(d2) < Math.PI ? 1 : -1);
                return [arc.centroid(d2), outerArc.centroid(d2), pos];
            };
        });

    polyline.exit()
        .remove();
};


</script>


</figure>


      </article>

   </center>

  </header>
</div>

<div class="wrapper row3">
  <main class="hoc container clear">

<center>

</center>

    <ul class="nospace group">

 <script>

      $.getJSON("tuits_positivos.json", function(data1){
          var html1 = [];

          /* loop through array */
          $.each(data1, function(index1, d1){
              html1.push("nombre : ", d1.nombre, ", ",
                      "texto : ", d1.texto, ", ",
```

```
                    "foto : ", d1.foto, "<br>");
                $("div6").append( '<img src=" ' + d1.foto + ' "  height="62" width="62">'+'<b>'+
d1.nombre +'</b>'+ ": " + d1.texto + '<br>' );

            });

            // $("div6").append(d.texto + " ");


        }).error(function(jqXHR, textStatus, errorThrown){ /* assign handler */
            /* alert(jqXHR.responseText) */
            alert("error occurred!");
        });

    </script>
    <script>

        $.getJSON("tuits_negativos.json", function(data2){
            var html2 = [];

            /* loop through array */
            $.each(data2, function(index2, d2){
                html2.push("nombre : ", d2.nombre, ", ",
                    "texto : ", d2.texto, ", ",
                    "foto : ", d2.foto, "<br>");
                $("div7").append( '<img src=" ' + d2.foto + ' "  height="62" width="62">'+'<b>'+
d2.nombre +'</b>'+ ": " + d2.texto + '<br>' );

            });

            // $("div6").append(d.texto + " ");


        }).error(function(jqXHR, textStatus, errorThrown){ /* assign handler */
            /* alert(jqXHR.responseText) */
            alert("error occurred!");
        });

    </script>
    <script>

        $.getJSON("tuits_neutros.json", function(data3){
            var html3 = [];

            /* loop through array */
            $.each(data3, function(index3, d3){
                html3.push("nombre : ", d3.nombre, ", ",
                    "texto : ", d3.texto, ", ",
                    "foto : ", d3.foto, "<br>");
                $("div8").append( '<img src=" ' + d3.foto + ' "  height="62" width="62">'+'<b>'+
d3.nombre +'</b>'+": " + d3.texto + '<br>' );
```

```
            });

            // $("div6").append(d.texto + " ");


        }).error(function(jqXHR, textStatus, errorThrown){ /* assign handler */
            /* alert(jqXHR.responseText) */
            alert("error occurred!");
        });

    </script>

    <div id="todo">
   <center><h1>TUITS</h1></center>
   <div6 id="izquierda"><p><b>POSITIVOS</b></p></div6>
   <div7 id="derecha"><p><b>NEGATIVOS</b></p></div7>
   <div8 id="contenido"><p><b>NEUTROS</b></p></div8>
</div>

    </ul>

    <div class="clear"></div>
  </main>
</div>

<div class="wrapper row3">

  <section class="hoc container clear">

  </section>
</div>
</div>
<div class="wrapper row5">
  <div id="copyright" class="hoc clear">
    <p class="fl_left">Copyright &copy; 2017 - All Rights Reserved - <a
href="#">Sentimientos</a></p>
    <p class="fl_right">POWERED by <a target="_blank" title="Equipo #1">Ezra
Saucedo</a></p>

  </div>
</div>

<a id="backtotop" href="#top"><i class="fa fa-chevron-up"></i></a>

<script src="/css/scripts/jquery.min.js"></script>
<script src="/css/scripts/jquery.backtotop.js"></script>
<script src="/css/scripts/jquery.mobilemenu.js"></script>

</body>
</html>
```