Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

# Aplicación de técnicas de minería de datos en redes sociales/web

Trabajo Fin de Máster

**Máster Universitario en Gestión de la Información**

**Autor**: Asensio Blasco, Elena

**Tutor**: Ferri Ramírez, César

[Curso 2014 - 2015]

# Resumen

Las técnicas de minería de datos permiten obtener información de redes sociales como Twitter. Su análisis correcto proporciona un valor adicional a la recuperación de información. El procesado de lenguaje natural, así como la delimitación geográfica del origen de sus mensajes, se convierte en un objetivo fundamental y punto de partida de cualquier investigación. Por ello, nuestro estudio se basa en la realización de una aplicación que gestione y obtenga datos sobre el uso de los lenguajes oficiales de las comunidades autonómicas en Twitter.

**Palabras clave:** Twitter, minería de datos, recuperación de información, geolocalización, procesado de lenguaje natural.

# Resum

Les tècniques de mineria de dades permeten obtenir informació de xarxes socials com Twitter. La seua anàlisi correcta proporciona un valor addicional a la recuperació d'informació. El processat de llenguatge natural, així com la delimitació geogràfica de l'origen dels seus missatges, es converteix en un objectiu fonamental i punt de partida de qualsevol investigació. Per això, el nostre estudi es basa en la realització d'una aplicació que gestione i done dades sobre l'ús dels llenguatges oficials de les comunitats autonòmiques a Twitter.

**Paraules clau**: Twitter, mineria de dades, recuperació d'informació, geolocalització, processament de llenguatge natural.

# Abstract

Data mining techniques obtain information from social networks like Twitter. The correct analysis of this data can provide useful and valuable knowledge. Natural language processing and geolocation of messages have become fundamental tools and a starting point of many investigations. This study addresses the development of an application to obtain and analyse Twitter data about the use of the official languages of the autonomous regions of Spain.

**Keywords :** Twitter, data mining, natural language processing, data science, geolocation, big data.



# Agradecimientos

Quisiera agradecer a mi familia y mis amigos por todo el cariño y comprensión en estos meses, y durante todos mis estudios. Principalmente a mi marido, gracias por apoyarme.

En segundo lugar, tanto a compañeros como a profesores de la UPV, gracias por formarme como persona y como profesional. En especial a mi tutor D. César Ferri Ramírez, por su orientación en este trabajo final de máster.

# Tabla de contenidos

## *Tabla de Ilustraciones*

# 1. Introducción

Las redes sociales se pueden definir como estructuras donde las personas mantienen algún tipo de relación. Dado que Internet se ha convertido en una herramienta fundamental en la comunicación, las relaciones del mundo real han pasado a lo virtual. Compartir información o simplemente charlar son las bases de cualquier red social.

Hay muchos campos donde la aplicación de redes sociales ha supuesto un enfoque nuevo, tales como los negocios, las investigaciones, la educación, el bien social o la comunicación, entre otras.

Si nos enfocamos a la red social Twitter, ésta permite la comunicación rápida y fácil con más de 500 millones de personas registradas, y más de 100 millones participando activamente al mes. Su popularidad ante el público, su bien documentada API o su inmediatez en tiempo casi real son las características que han hecho que nos decantemos por esta red social.

El objetivo del proyecto consiste en aplicar técnicas que permitan obtener datos recopilados desde la red social Twitter. Para ello hemos realizado una aplicación que obtiene las coordenadas y el idioma de los mensajes en twitter.

Nuestros objetivos con este proyecto son:

- ✓ La recopilación de tweets a través del API que proporciona Twitter.
- ✓ La clasificación de los tweets según su zona geográfica. En concreto, la obtención de tweets geolocalizados en territorio nacional.
- ✓ El análisis lingüístico de los textos de Twitter de forma automática, siendo prioridad principal la obtención de tweets en las distintas lenguas oficiales.
- ✓ Proporcionar una visualización significativa de los datos obtenidos.

Entre sus posibles utilidades estaría estudiar la predominancia lingüística por comunidad autónoma, municipio e incluso entre barrios; estudiar focos de turistas en un determinado lugar, el uso de idiomas en eventos o festivales, o aproximarse más al usuario adecuando la publicidad a su idioma.

Por tanto, en esta memoria vamos a tratar los temas:

- ❖ Analizar qué información podemos obtener del API de Twitter.
- ❖ Qué es el API Rest y la streaming API: Las herramientas que proporciona twitter para obtener los datos.
- ❖ Twitter y la geolocalización: Cómo recoger la cantidad máxima de tweets de una determinada región.
- ❖ Clasificación de los tweets por idioma, en concreto la diferenciación entre los lenguajes oficiales de las comunidades autónomas.
- ❖ Visualización de los tweets en un mapa.

# 2. Trabajos relacionados

En muchos campos, el estudio de redes sociales como herramientas de obtención de datos, ha supuesto un gran avance. Twitter ha abierto nuevas oportunidades de investigación y de negocio.

Uno de los temas más interesantes es el análisis de sentimiento y de opinión, donde se obtienen si los textos de los tweets contienen un sentimiento positivo, neutro o negativo. En este artículo [1] además de palabras o expresiones de felicidad o de tristeza, buscan emoticonos del usuario y les asignan valor positivo o negativo:

- Emoticonos felices: “:-)”, “:)”, “=)”, “:D” etc.
- Emoticonos tristes: “:-(”, “:(”, “=(”, “;(” etc.

Después de analizar el texto, éste determina si tiene más palabras con un sentimiento u otro, dado así su resultado final.

En cuanto al análisis de datos geolocalizados, vamos a ver cuatro herramientas de interés:

**Tweepsmap** [2]. Ubica los seguidores de una cuenta en un mapa. Para acceder a los datos de la tienes que lanzar un tweet con un resumen de las estadísticas de la cuenta o las características de la herramienta. Permite analizar por país, provincia o ciudad, e incluye gráficos y listados.

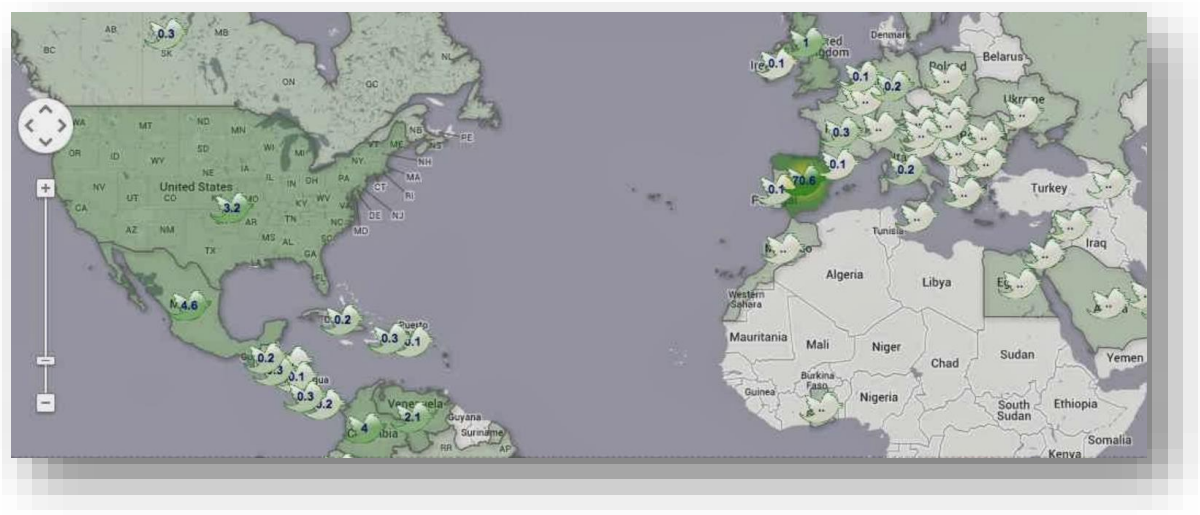


*Ilustración 1. Captura de un ejemplo de Tweepsmap*

**Trendsmap** [3]. Geolocaliza las tendencias en tiempo real de cualquier lugar del mundo. También puedes seleccionar un tema en concreto y saber dónde se está hablando. En la versión de pago se añaden filtros por usuario, palabras clave o idioma. Además de las tendencias, links, fotos y usuarios más influyentes.

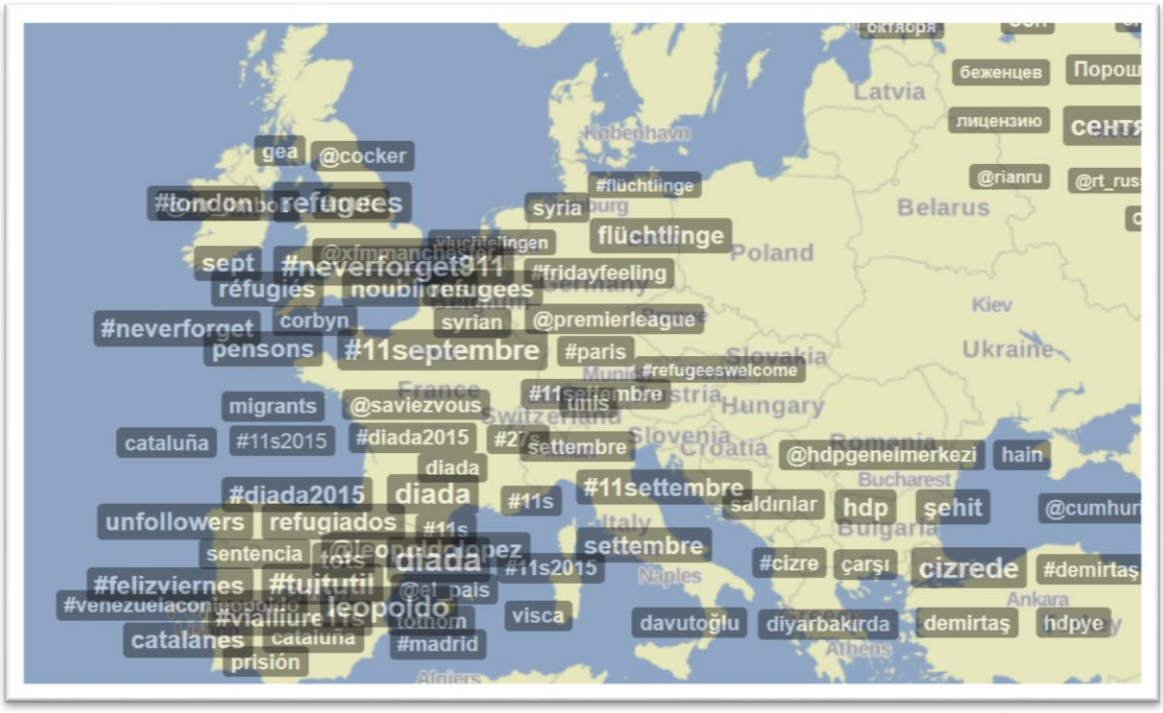


*Ilustración 2. Captura de la herramienta Trendsmap*

**Twaps** [4]. Busca los usuarios de Twitter y los 100 últimos tweets generados en un radio de dos millas alrededor de una localización indicada. En el mapa aparecen el avatar del usuario, donde al hacer clic, se accede a información sobre su nombre, su tweet, la fecha y la hora.

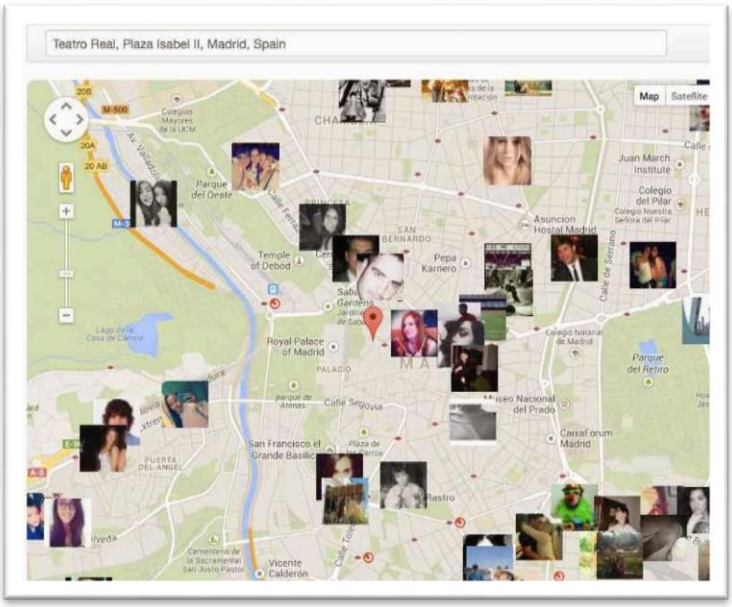


*Ilustración 3. Captura de un ejemplo de Twaps*

**The One Million Tweet Map** [5]. Geolocaliza en un mapa del mundo el último millón de tweets publicados, y se va actualizando en tiempo real. Otra de sus opciones interesantes es que también se puede filtrar por palabras clave o con hashtags, y ver dónde hablan de ese tema. Al hacer zoom en el mapa, también se amplía el foco de actividad sobre una localización concreta.

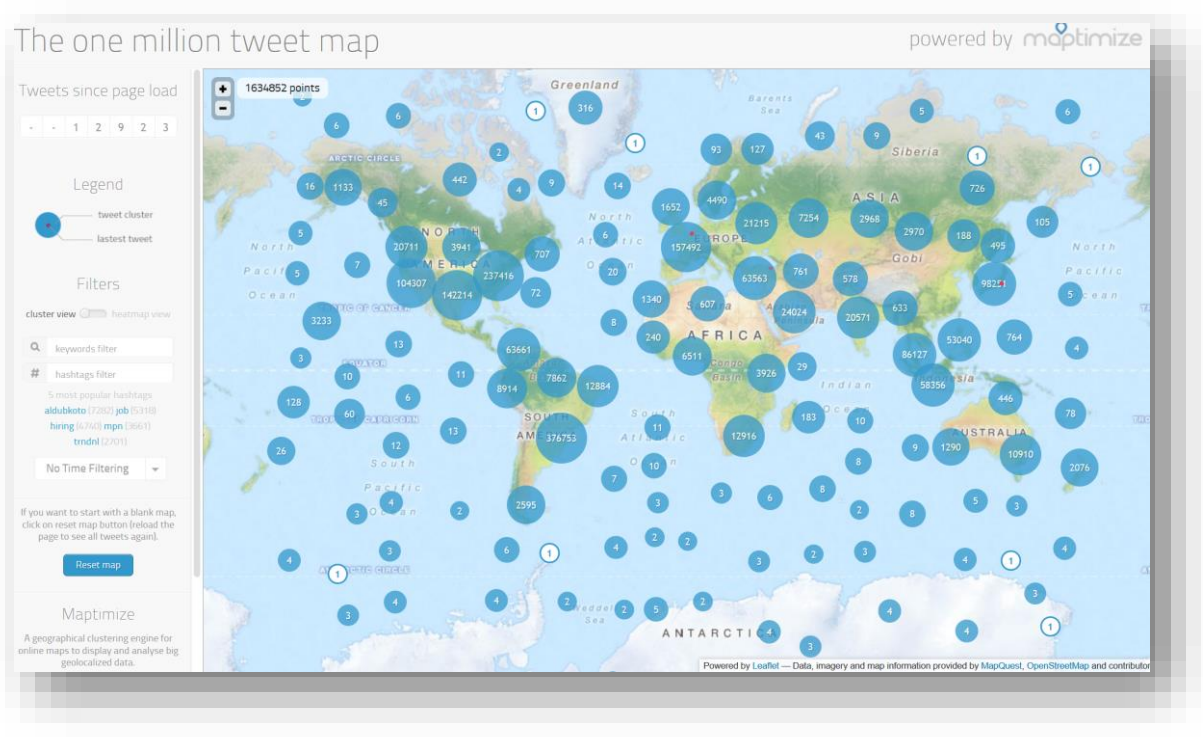


*Ilustración 4. Captura de la herramienta The One Million Tweet Map*

# 3. Twitter y su API

## ▪ Qué es Twitter

Twitter es un servicio de microblogging que permite enviar mensajes de texto con un máximo de 140 caracteres. Estos mensajes se llaman tweets, y aparecen en la página principal del usuario. Cada usuario puede seguir a otros usuarios y ver sus tweets. Se puede acceder a Twitter desde la web (www.twitter.com), con aplicaciones para smartphones, o a través de SMS en ciertos países.


*Ilustración 5. Vista de la cuenta de Twitter de la UPV desde la web*

## ▪ Glosario de términos en Twitter

- Tweet (*Tuit*): Publicación o actualización en Twitter.
- Followers (*Seguidores*): Usuarios que siguen una cuenta y leen los tweets que se envían.
- Following (*Seguidos*): Son las cuentas de Twitter que un usuario sigue. Los tweets publicados en las cuentas aparecen automáticamente en el timeline del usuario.
- Timeline (*TL*): Lista de tweets enviados de las cuentas que se siguen de manera cronológica.

- Retweet (*RT o retuit*): Función que permite volver a publicar un tweet, citando al usuario autor.

## Qué información podemos extraer de Twitter

Podemos dividir en cuatro secciones lo más relevantes de Twitter:

- **Quién**. La persona que lo escribió, o hizo un retweet, junto con sus datos públicos: nombre completo, localización, lenguaje, etc.
- **Cuándo.** Fecha y hora de publicación. Por ejemplo, con el perfil del usuario se puede determinar el horario en el que se encuentra.
- **Qué**. El contenido del tweet, que incluye el texto del mensaje además de los links, menciones o contenido multimedia.
- **Dónde**. Coordenadas geográficas de la ubicación desde donde fue publicado. Hay que tener en cuenta que no aparece en todos los tweets, sino que esta información es opcional.

En nuestro caso, es importante que podamos extraer solamente tweets que cuenten con datos geográficos.

## Información extraíble de la API de Twitter

Twitter proporciona su propio API (Application Programming Interface) oficial. Un API permite la comunicación entre diferentes componentes de software, añadiendo una capa entre ellos. El API de Twitter permite controlar tu cuenta y recuperar la información desde código. Este API se encuentra actualmente en su versión 1.1.

Hay dos maneras de extraer información desde Twitter: la API Rest y la Streaming API.

- **API Rest**. Permite realizar todas las acciones a las que tenemos acceso desde la página web o las aplicaciones. Proporciona acceso a la información ya existente en Twitter en el momento de hacer la llamada. Accedemos a los datos por un sistema en forma de caja negra realizando peticiones GET y POST.

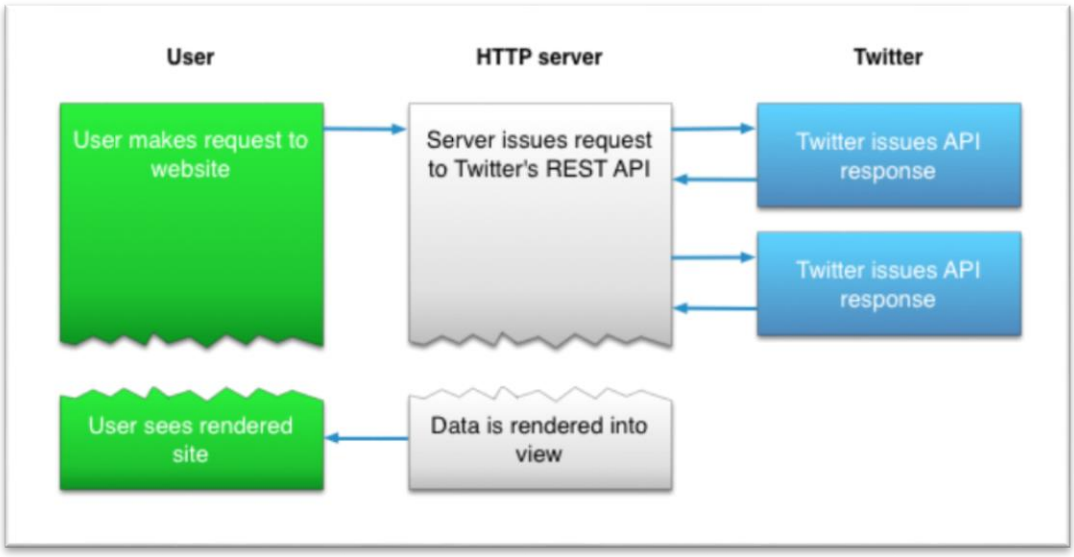


*Ilustración 6. Esquema del funcionamiento de la API Rest de Twitter*

- **Streaming API**. Recibiremos información creada posteriormente a la petición de datos. Se inicia con Twitter abriendo una conexión entre su servidor y nuestro sistema, y enviará por ella tweets que sean publicados a partir de ese momento, siempre que cumplan los filtros indicados al inicio de la conexión hasta su cierre. Es decir, es un proceso en tiempo real.

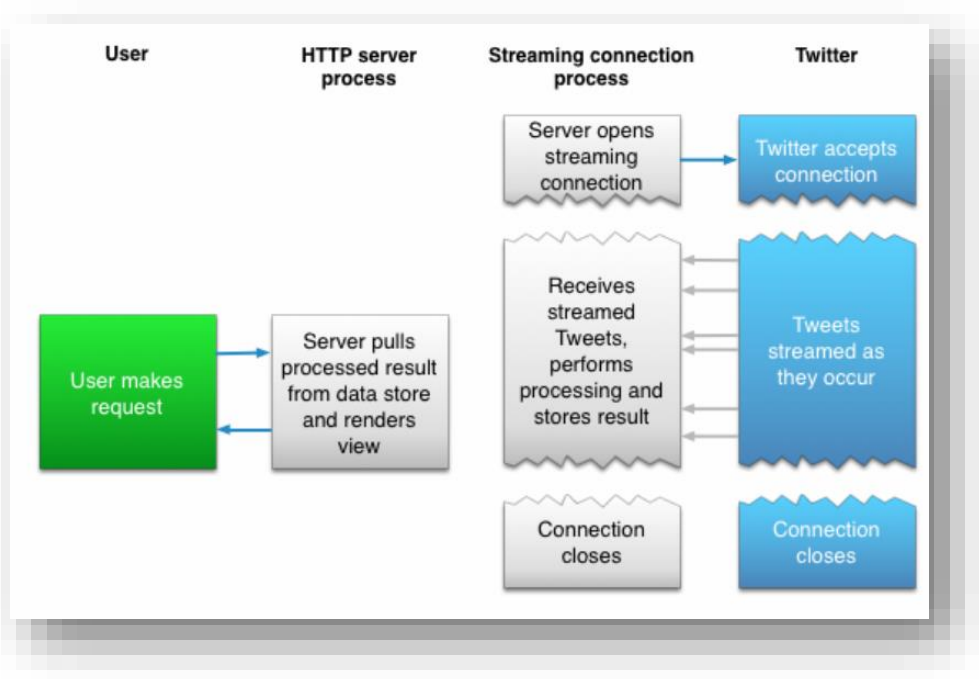


*Ilustración 7. Esquema del funcionamiento del API Streaming de Twitter*

**Formato de salida JSON**

El formato para representar la información en Twitter es JSON[6], *JavaScript Object Notation*, es un formato sencillo orientado para el intercambio de datos.

Una de las ventajas de JSON como formato de intercambio de datos es su simplicidad para escribir un analizador sintáctico, también llamado *parser* de JSON. Un ejemplo de salida de datos sería este fragmento de una búsqueda de tweets.

```
{
  "statuses": [
    {
      "coordinates": null,
      "favorited": false,
      "truncated": false,
      "created_at": "Mon Sep 24 03:35:21 +0000 2012",
      "id_str": "250075927172759552",
      "entities": {
        "urls": [

          (…)
```

Desde la versión 1.1 del API de Twitter, necesitaremos una autentificación OAuth[7] si queremos realizar nuestra propia aplicación.

**Protocolo de seguridad OAuth**

OAuth [8] *(Open Authorization)* es un protocolo que permite autorización segura de una API de un modo estándar y simple para aplicaciones de escritorio, móviles y web. Este proporciona a los usuarios un acceso a sus datos al mismo tiempo que protege las credenciales de su cuenta.

- Pasos para obtención de los tokens

Primero, entraremos en la siguiente dirección:

https://dev.twitter.com/

Allí entramos con nuestra cuenta de usuario de Twitter, y después visitaremos:

https://apps.twitter.com/

Donde crearemos una aplicación nueva e introduciremos los siguientes datos:

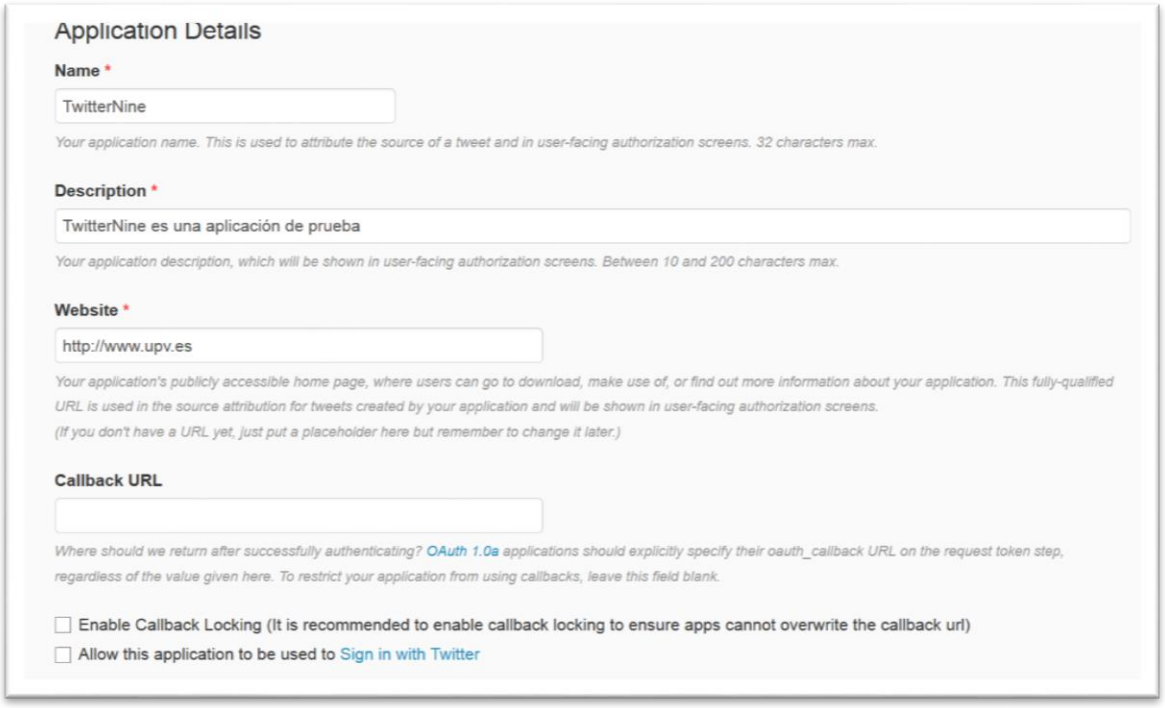


*Ilustración 8. Formulario a completar para la aplicación de Twitter*

Una vez registrada, necesitamos los valores: *Consumer Key (API Key)*, *Consumer Secret (API Secret)*, *Access Token* y *Access Token Secret* que se encuentran en la pestaña *Keys and Access Tokens* y le damos al botón *Create my access token*. Estos tokens son los que nos darán permiso para acceder a nuestra cuenta y debemos indicar en el código de la aplicación.

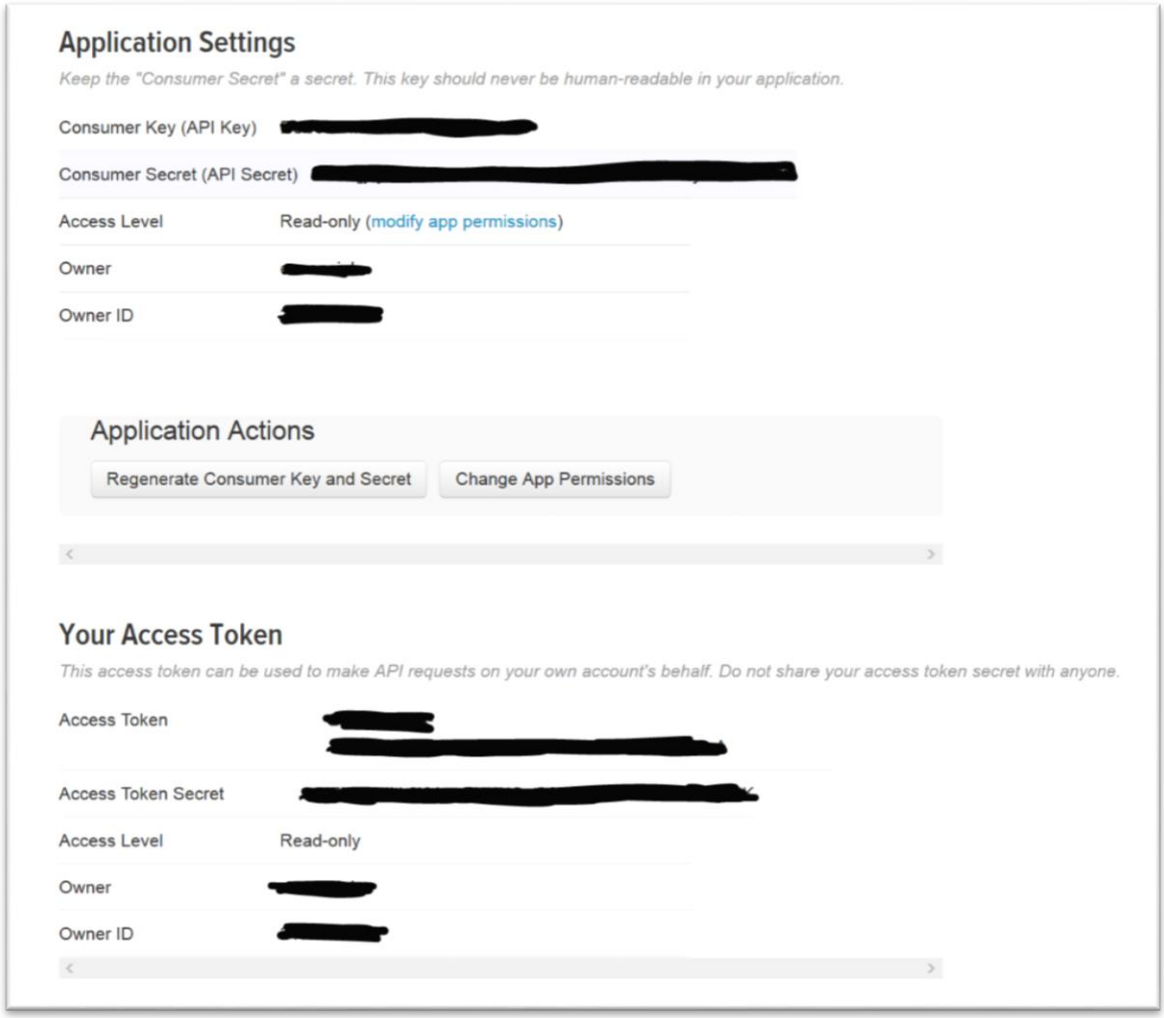


*Ilustración 9. Tokens de seguridad de la aplicación*

Si queremos limitar el flujo de datos de nuestra cuenta a la aplicación o viceversa, podemos dar permisos de sólo lectura. Para ello debemos ir a la pestaña de *Permissions*.

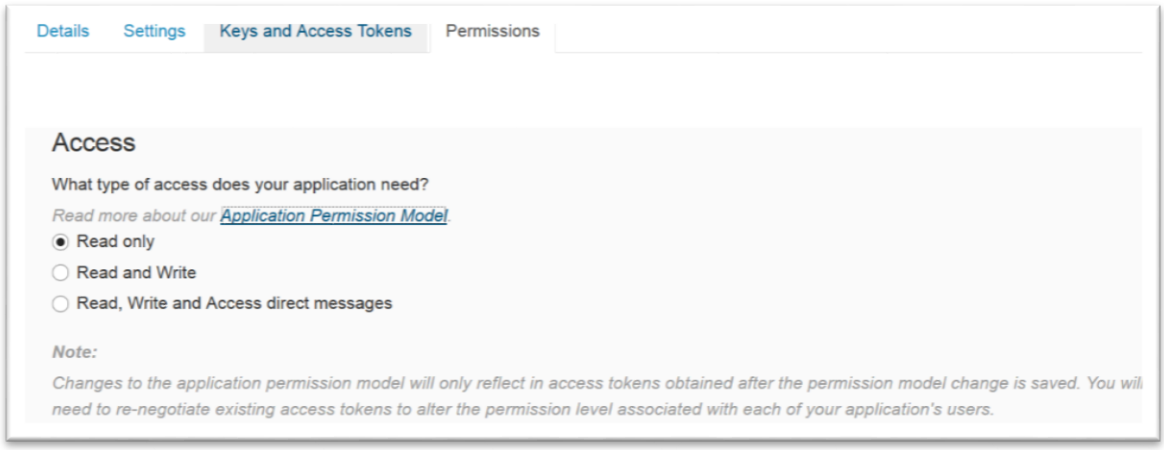


*Ilustración 10. Ajuste de permisos de nuestra aplicación*

- Librerías en Python para acceder a la API de Twitter

Adicionalmente a realizar peticiones GET y POST directamente desde código, existen librerías de carácter abierto que facilitan este tipo de acceso con métodos ya predefinidos.

**Tweepy**

Tweepy[9] es probablemente la librería open source más conocida para acceder a la API desde Python, provee acceso a los métodos de la API de Twitter. Se encuentra en github y tiene una documentación muy completa y bastantes ejemplos.

Para instalar Tweepy, la forma más sencilla es:

```
pip install tweepy
```

Si no tenemos instalado el programa `pip` podemos hacerlo con:

```
sudo easy_install pip
```

Por ejemplo, con Tweepy podemos sacar textos de los últimos 20 tweets de nuestro timeline:

```
import tweepy

#Funciones para autentificarnos en Twitter
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth)

#Guardamos los 20 tweets del timeline
public_tweets = api.home_timeline()

#Recorremos y mostramos por pantalla
for tweet in public_tweets:
    print tweet.text
```

En este otro ejemplo, podemos buscar un usuario por su nombre y sacar su lista de seguidores:

```
user = tweepy.api.get_user('twitter')

#Imprime el nombre del usuario por pantalla
print user.screen_name

#Imprime el número de seguidores por pantalla
print user.followers_count

#Imprime su lista de seguidores
for friend in user.friends():
```

```
print friend.screen_name
```

Otras librerías open source para trabajar con Python y Twitter son:

- Twitter API [10]. Soporta tanto el API Rest como Streaming API con OAuth 1.0 y OAuth 2.0, además de Python en sus versiones 2.X y 3.X.
- Python-Twitter [11]. Actualmente ofrece métodos para Python en 2.X y se encuentra en desarrollo en su versión de Python 3.X.

## ▪ Datos geolocalizados

Disponemos de tres maneras diferentes para clasificar los tweets geográficamente:

1. **Lugar del mensaje**. tweets que están marcados con la localización exacta. Puede ser la localización exacta o el 'Twitter Place'.
   - Localización exacta con coordenadas latitud/longitud: P.ej. -85.7629, 38.2267
   - El Twitter Place es un metadato que marca un lugar, p. ej. "Louisville Central", y cuatro pares de coordenadas latitud/longitud para definir el área indicada.

2. **Lugar del perfil.** Uno de los datos que puede indicar el usuario es la localidad donde vive, esta aparece en su perfil de manera pública.


*Ilustración 11. Ejemplo de perfil en Twitter con lugar en el perfil*

3. **Localización mencionada en el mensaje**. Mención de un lugar en el texto del tweet. P. ej. “La lluvia en Sevilla es una maravilla”.

Dada su exactitud, obtener los tweets del primer caso será prioritario, sin embargo sólo un 2% de los tweets publicados globalmente tienen este dato [12]. Ampliaremos nuestra muestra guardando también el lugar del perfil, si este contiene una localidad española.

Descartamos guardar datos sobre la localización según el texto, pues puede no ser indicador de que el usuario escribe desde ese lugar.

Para poder enlazar municipios con las coordenadas correspondientes se ha trabajado con un Excel que hemos adaptado. Este contiene todos los pueblos y ciudades de España [13] en el siguiente formato:

| Comunidad | Provincia | Población | Latitud | Longitud | Altitud | Habitantes | Hombres | Mujeres |
|---|---|---|---|---|---|---|---|---|
| Andalucía | Almería | Abla | 37,14114 | -2,780104 | 871,1684 | 1504 | 783 | 721 |
| Andalucía | Almería | Abrucena | 37,13305 | -2,797098 | 976,9387 | 1341 | 682 | 659 |
| Andalucía | Almería | Adra | 36,74807 | -3,022522 | 10,97898 | 24373 | 12338 | 12035 |
| Andalucía | Almería | Albánchez | 37,2871 | -2,181163 | 481,3123 | 815 | 422 | 393 |
| Andalucía | Almería | Alboloduy | 37,03319 | -2,62175 | 388,4346 | 674 | 334 | 340 |
| Andalucía | Almería | Albox | 37,38979 | -2,147483 | 426,4268 | 11178 | 5731 | 5447 |
| Andalucía | Almería | Alcolea | 36,97449 | -2,961038 | 744,6956 | 908 | 486 | 422 |
| Andalucía | Almería | Alcóntar | 37,33585 | -2,596944 | 955,1331 | 603 | 315 | 288 |
| Andalucía | Almería | Alcudia de Monteagud | 37,23598 | -2,266174 | 1018,354 | 141 | 74 | 67 |

*Ilustración 12. Formato del Excel con la lista de municipios españoles*

# 4. Herramientas y configuración

Se ha realizado todo el proyecto en el sistema operativo Ubuntu 14.04 de 32 bits, ya que es un entorno con Python preinstalado y más eficiente a la hora de instalar las herramientas que necesitamos.

El primer paso, fue configurar Eclipse Mars 4.5 para Python. Para ello, hemos añadido PyDev desde el menú *Help > Install New Software*. Seleccionamos las opciones que aparecen y las instalamos.

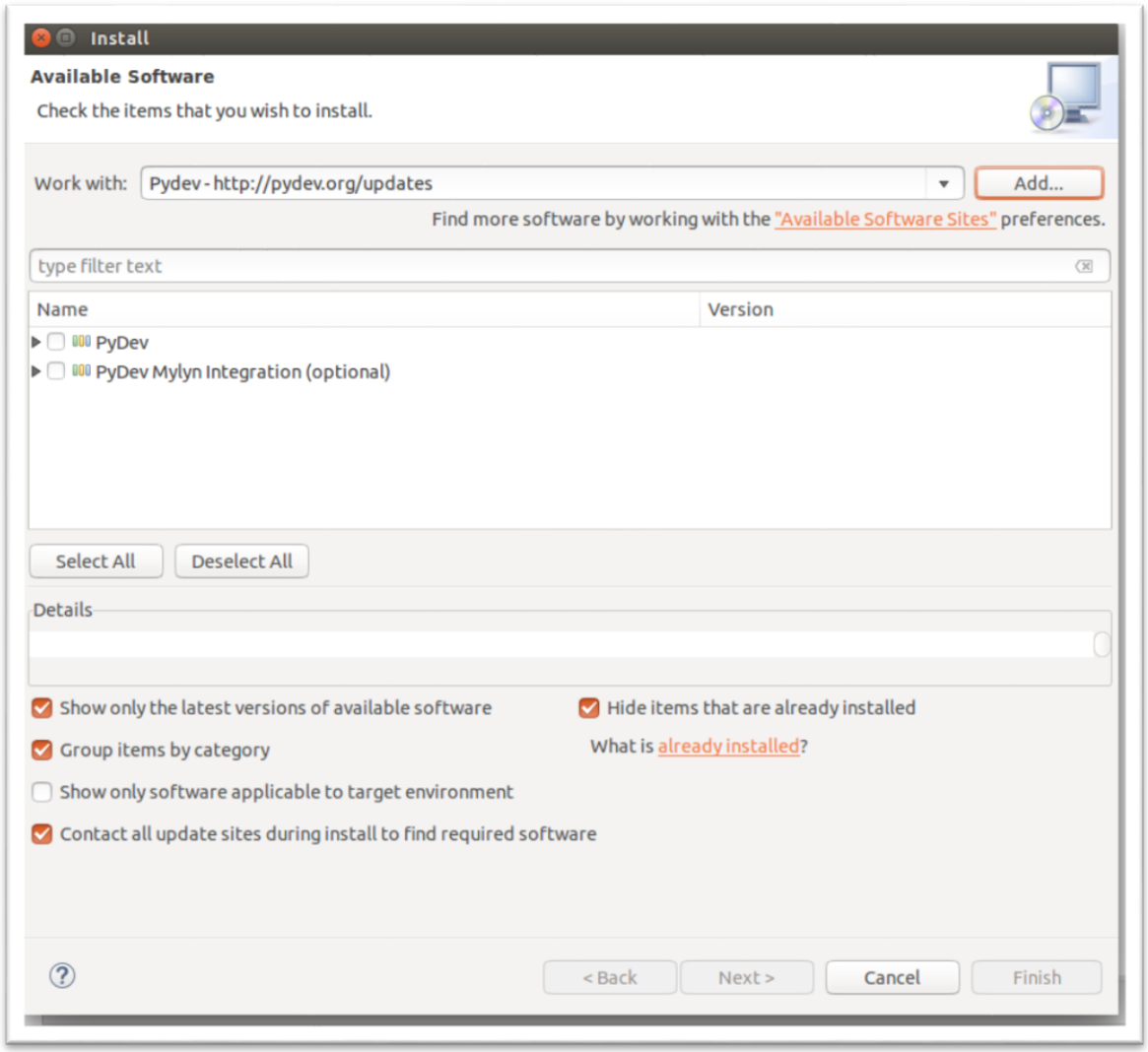


*Ilustración 13. Instalación en Eclipse del paquete Pydev para Python*

Si la instalación ha sido correcta, nos aparecerán los siguientes datos. A destacar que la versión de PyDev para Eclipse ha sido la 4.3.0

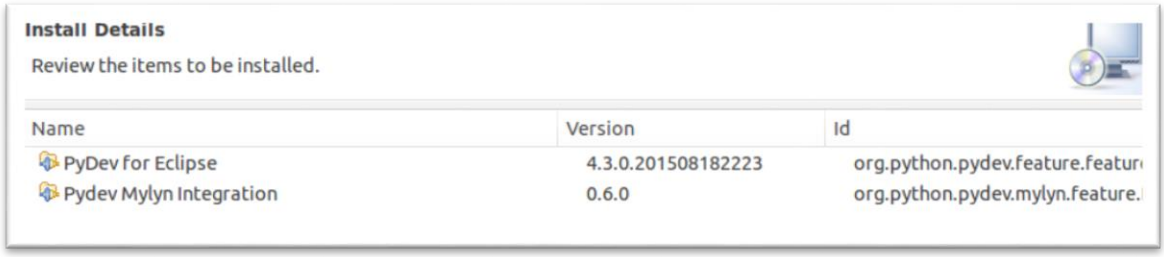


*Ilustración 14. Detalles de instalación correcta Pydev por Eclipse*

Una vez tenemos preparado nuestro IDE para Python, instalaremos la API Open Source Tweepy, y las otras librerías que necesita la aplicación:

- La librería Xlrd 0.9.4[14] para extraer datos de hojas de datos Excel.
- La librería JSON[15], que convierte los datos en formato JSON en una lista o un diccionario Python, o viceversa.

Por último necesitaremos instalar un intérprete de Python para ejecutar la aplicación. Vamos a *Windows>Preferences>PyDev>Interpreters>Python Interpreter>New*

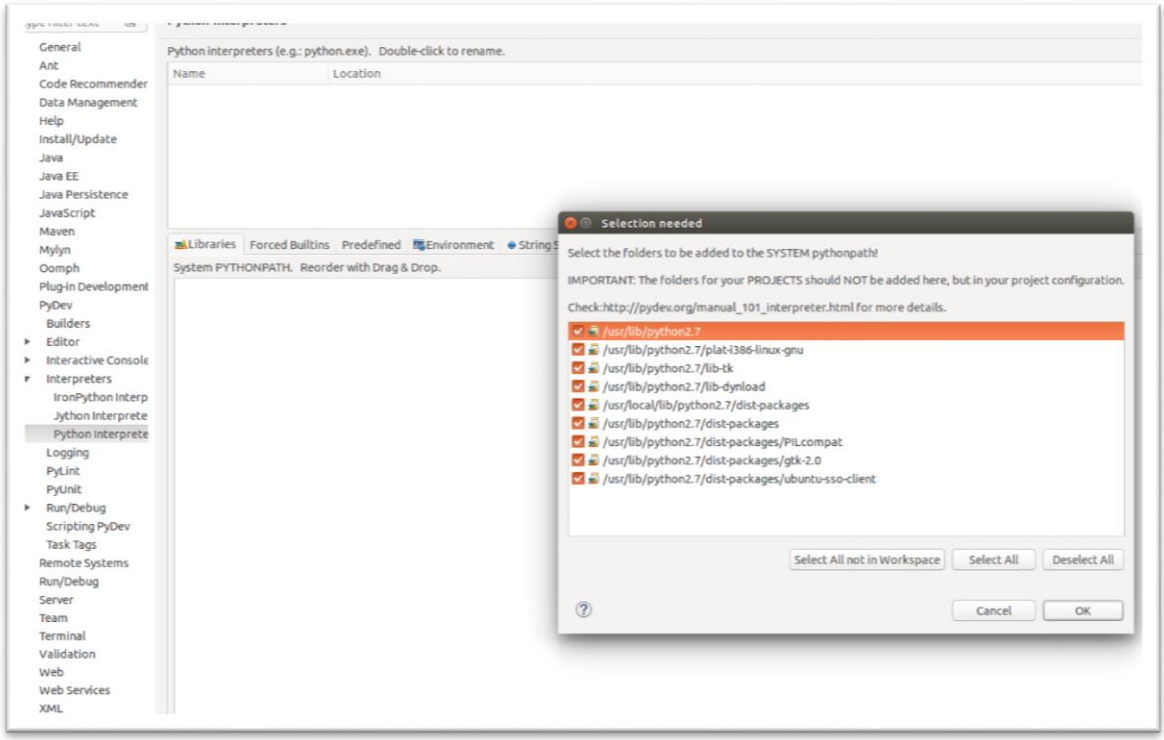


*Ilustración 15. Configuración del intérprete de Python en Eclipse*

Seleccionamos la versión de Python 2.7, que es la que usamos en la aplicación y la añadimos al proyecto.

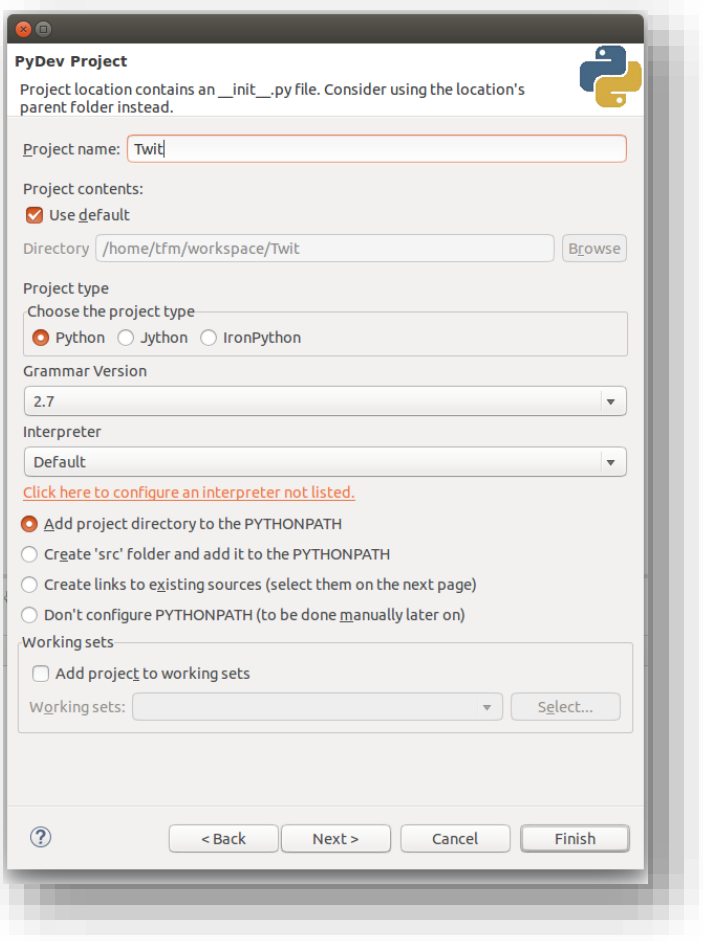


*Ilustración 16. Configuración del intérprete y adición al proyecto*

En orden de mostrar los resultados de una manera más directa, se ha realizado una página web sencilla en php, *geoWebTwit.php*. Para ello, necesitamos tener un servidor donde se muestre, instalaremos Apache en nuestro caso por terminal.

```
sudo apt-get install apache2
```

Ahora tendremos la ruta /var/www/ preparada para guardar nuestro archivo en php o realizar un enlace simbólico al archivo. Así mostraremos nuestra página web en la siguiente dirección del navegador:

```
localhost/geoWebTwit.php
```

# 5. Clasificación de Tweets por idioma

Uno de los puntos a tratar en este trabajo, es la diferenciación por idioma de los tweets. Para ello utilizaremos un kit de herramientas de lenguaje natural, llamado NLTK *(Natural Language ToolKit)*. Este conjunto de bibliotecas para el procesamiento del lenguaje natural se puede utilizar en Python para identificar el idioma del texto.

Para instalar este módulo, pondremos las siguientes instrucciones en el terminal:

```
sudo pip install -U nltk
```

```
sudo pip install -U numpy
```
(Opcional)

También es muy importante instalar la base de datos NLTK Data que proporciona archivos con textos históricos, palabras y stopwords

```
python -m nltk.downloader all
```

Si queremos asegurar la instalación completa:

```
sudo python -m nltk.downloader -d /usr/share/nltk_data all
```
(Opcional)

Vamos a detectar el idioma partiendo de una cadena de texto almacenada en una variable en Python[16]. Para ello haremos uso de las stopwords, también llamadas *palabras vacías*, que son palabras que no tienen ningún significado por sí mismas, como las preposiciones o los artículos.

Se utilizan las stopwords ya que pueden encontrarse un grupo de ellas en la mayoría de los idiomas, y aunque no dan significado a las frases, permiten la unión de palabras entre ellas, algo importante en el lenguaje natural.

En definitiva, recorreremos el texto palabra a palabra, contando el número de stopwords que se encuentran para cada idioma. El idioma con más palabras en común con el texto será el propuesto por la aplicación.

```
def idiomaTexto(texto):

    #Lista de idiomas disponibles en la nltk
    languages=["spanish","english","catala", "basque",
"galician"]
    #Texto a analizar
    text_to_detect = texto
    #Dividimos el texto de entrada en tokens o palabras únicas
    tokens = nltk.tokenize.word_tokenize(text_to_detect)
```

```
    tokens = [t.strip().lower() for t in tokens] # Convierte
todos los textos a minúsculas para su posterior comparación
    #Creamos un diccionario donde almacenaremos la cuenta de las
stopwords para cada idioma
    lang_count = {}
    #Por cada idioma
    for lang in languages:
        #Obtenemos las stopwords del idioma del módulo nltk
     stop_words = unicode(nltk.corpus.stopwords.words(lang))
     lang_count[lang] = 0 #Inicializa a 0 el contador para cada
idioma
     #Recorremos las palabras del texto a analizar
     for word in tokens:
          if word in stop_words: #Si la palabra se encuentra
entre las stopwords, incrementa el contador
               lang_count[lang] += 1
    #Obtiene el idioma con el número mayor de coincidencias
    detected_language = max(lang_count, key=lang_count.get)
    return detected_language
```

Algunos de los idiomas que soporta NLTK por defecto son: inglés, español, alemán o portugués. Debido a ello, hemos añadido los stopwords del gallego [17], y también catalán y euskera de Lucene 5.3.0. La lista de stopwords del gallego fue editada puesto que el formato admitido era UTF-8, y se tuvieron que eliminar algunas stopwords que contenían la 'ñ'. El directorio donde se encuentran es *./usr/share/nltk_data/corpora/stopwords* tal y como aparece en la captura.


*Ilustración 17. Captura de la ubicación de los stopwords en NLTK_DATA*

Hay que tener en cuenta, que a mayor número de palabras analizadas, más acierto en el idioma del texto. Es posible que en casos de tweets con poco texto, con varias palabras en otro idioma, o nombres propios, el resultado dado difiera con el idioma real.

# 6. Visualización de Tweets

Nuestra prioridad a la hora de tratar los tweets, es obtener tanto el idioma del texto, como su localización, si esta se corresponde a un lugar del territorio español. El formato de un tweet una vez analizado consistirá en lo siguiente:

TWEET Num.13
Texto mensaje: Acto de apertura @UPV 2015: discurso de Juan Ignacio Cirac https://t.co/aAOS8ojB1h
Idioma del tweet: spanish
Lugar del tweet: None
Lugar del usuario: Alcoy, Gandia, Valencia
Coordenadas del usuario: 38.69763, -0.4730959

Nos hemos enfocado a recopilar la información del propio Timeline del usuario, ya que hay menores limitaciones de peticiones y tiempo. Por ejemplo, el máximo número de peticiones de búsqueda son 180 cada 15 minutos. Al estar autentificadas las llamadas, no se cuentan por su IP [18]. Además la cantidad de tweets que se obtienen cada vez que se ejecuta la aplicación se ha fijado a 20.

Luego de sacar los datos por pantalla, guardamos tanto las coordenadas como el idioma en un archivo csv llamado *geotuits.csv*. El CSV [19] es un tipo de documento sencillo y de formato abierto para representar datos en forma de tabla, separando por comas sus columnas y saltos de línea sus filas. Será de este archivo de datos del cual leeremos en nuestra web *geoWebTwit.php*.

La página web muestra un mapa interactivo realizado con la librería Leaflet [20], una librería Open Source realizada en JavaScript, en su versión 0.7.5. En esta captura, vemos como están marcados varios lugares de la geografía, e indica el idioma en el cual se escribió el tweet al hacer clic sobre el marcador.

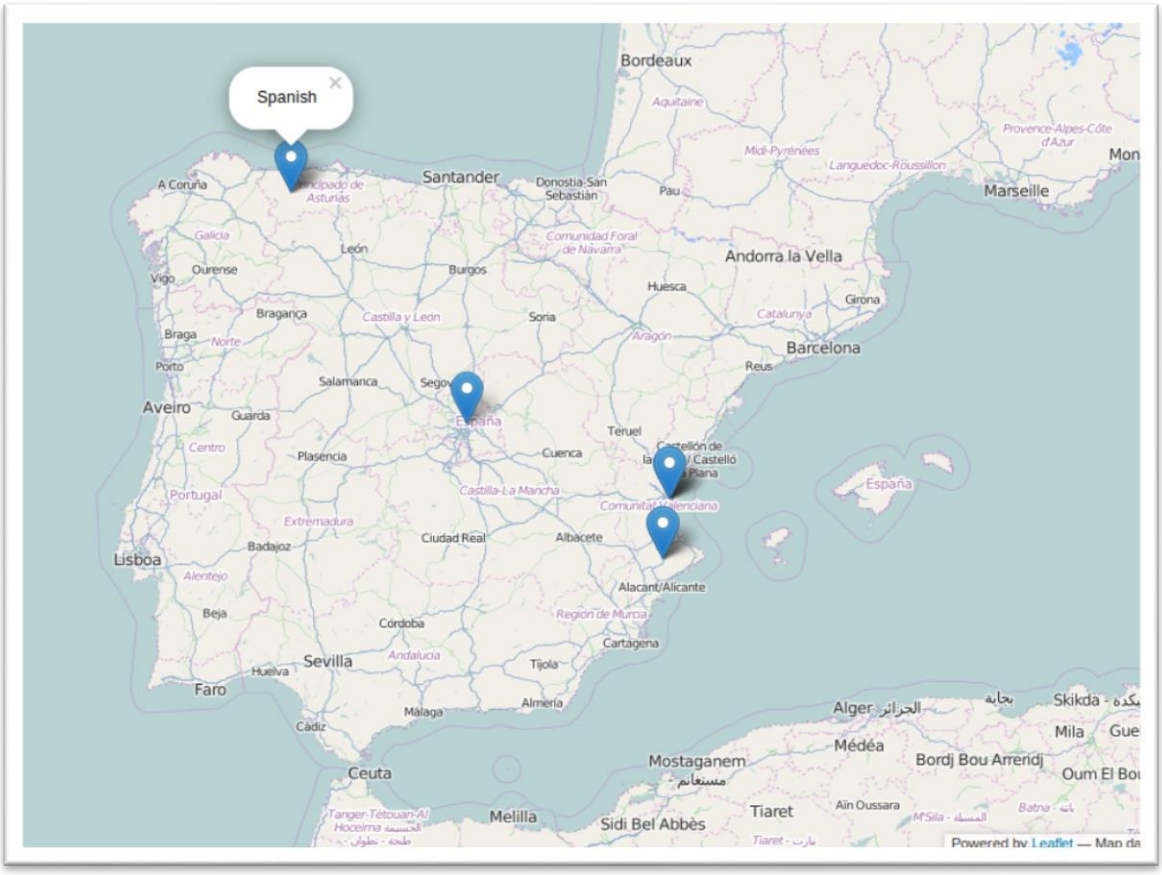


*Ilustración 18. Captura de la web de la aplicación*

## Estadísticas y resultados

Dado que nuestro análisis se basa en los tweets que obtenemos a través de la aplicación, mostraremos una serie de estadísticas basadas en nuestros resultados. Resaltar que según el usuario analizado podrían variar, ya que el Timeline cambia de forma sustancial dependiendo de cada perfil.

Vemos en este gráfico que poco más de la mitad de los tweets muestran algún dato de localización. De un total de 1200 tweets, tenemos 781 con un lugar asociado y 419 sin municipio o país. En este número se observan tanto tweets con coordenadas como lugar en el perfil del usuario.

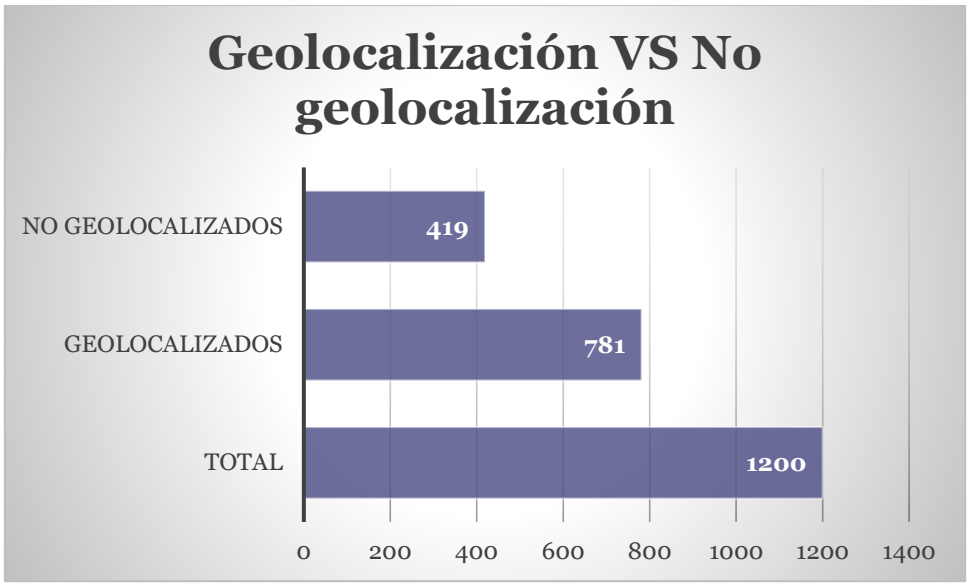


*Ilustración 19. Clasificación de Tweets: Geolocalización*

Podemos sacar en claro que en general, los porcentajes son los siguientes: 65% geolocalizados frente a 35% no geolocalizados.

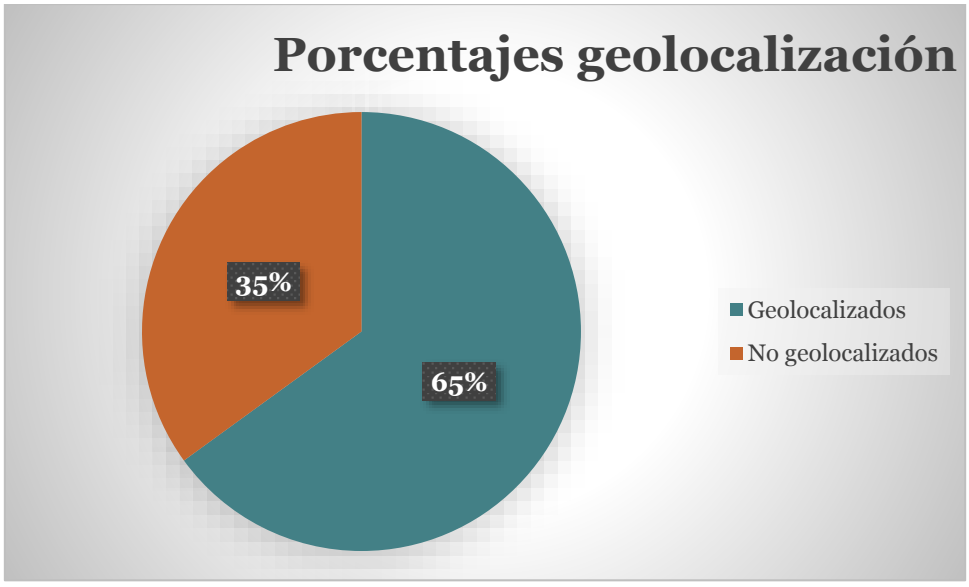


*Ilustración 20. Porcentajes geolocalización totales*

Dentro de los tweets geolocalizados, debemos dividir entre los tweets mandados desde España y los de fuera de ella. Vemos que sólo 473 son geolocalizados nacionalmente.

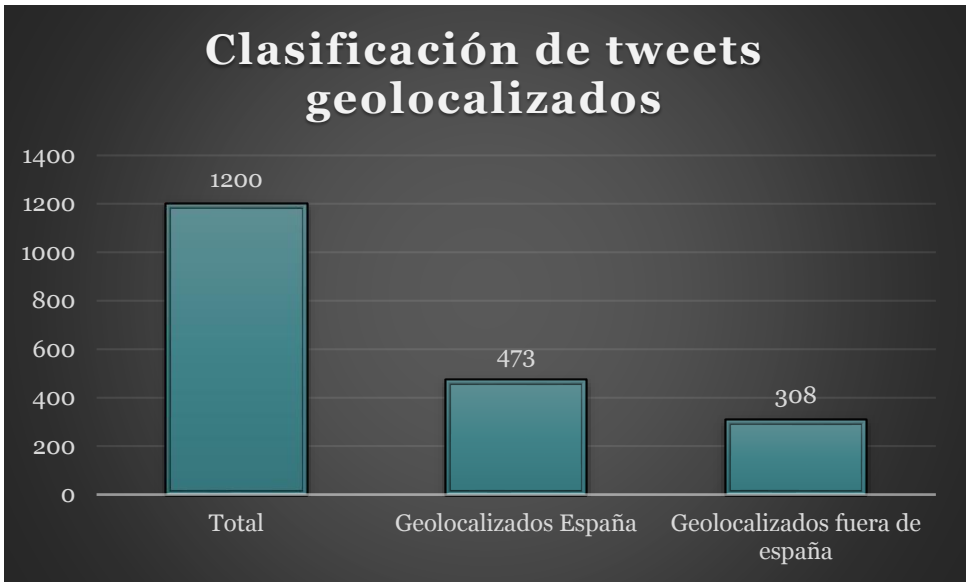


*Ilustración 21. Clasificación de Tweets geolocalizados*

En porcentaje, vemos que dentro de los tweets geolocalizados, la mayor parte son desde España, aproximadamente 61% frente al 39% extranjero.

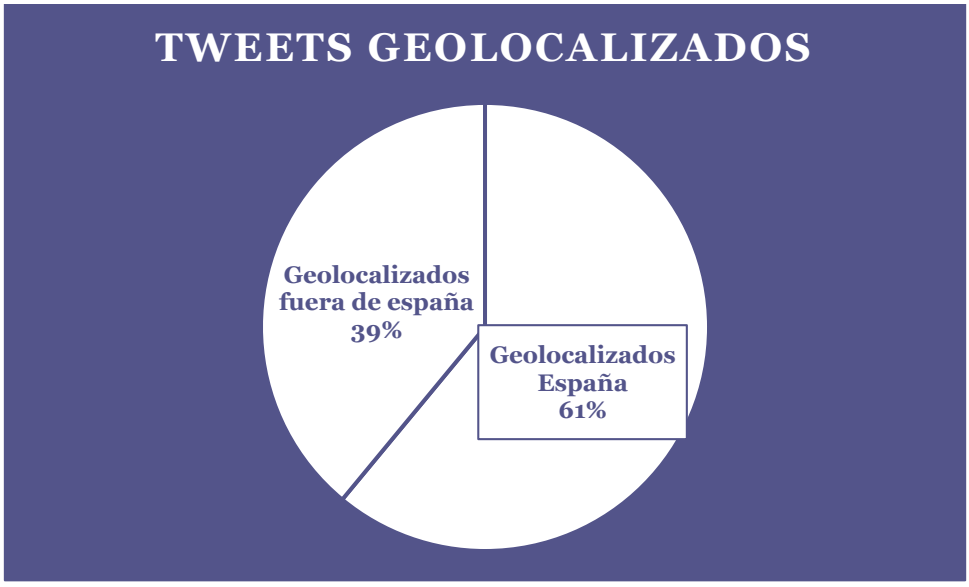


*Ilustración 22. Clasificación de Tweets geolocalizados en España*

En el tema del idioma, analizamos los siguientes idiomas: español, inglés, catalán, gallego y euskera. En la muestra, no encontramos tweets en gallego en este caso. Obtenemos 487 tweets en español, 432 en inglés, 163 en catalán y 118 en euskera.

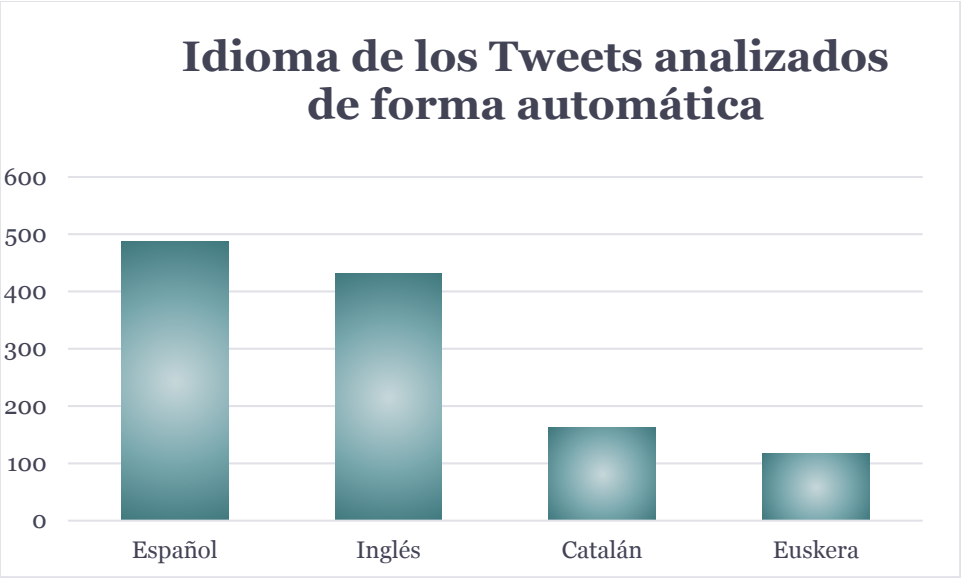


*Ilustración 23. Gráfico con el idioma de los tweets analizados*

Puesto que la identificación del idioma se efectúa de manera automática, hemos tomado una muestra simple de 20 tweets que se puede ver íntegramente en el apéndice 1. Esta se ha revisado a mano para corregir el idioma. El porcentaje de aciertos se sitúa en el 70%.

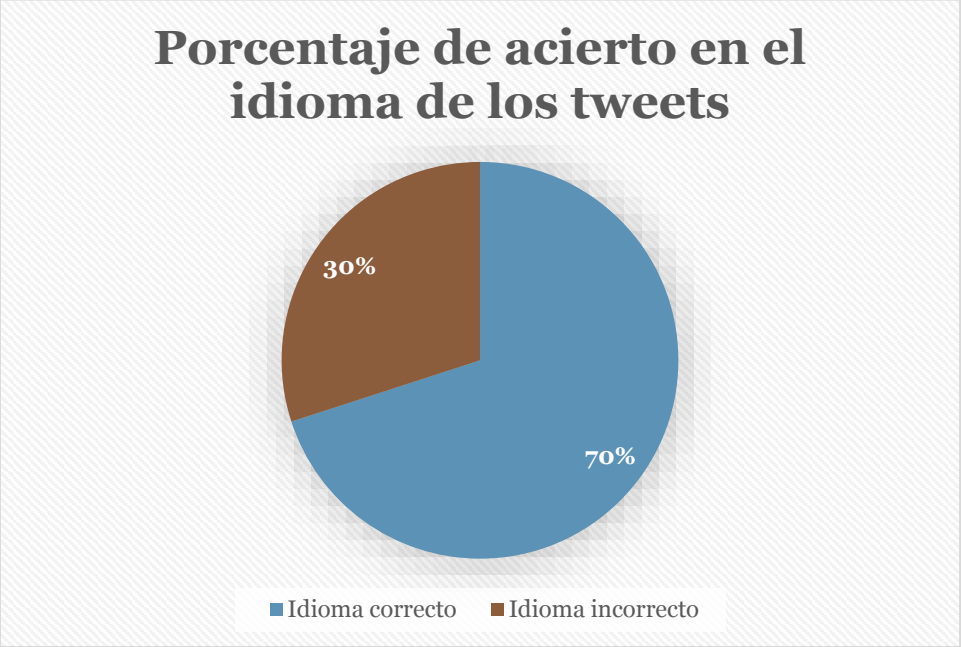


*Ilustración 24. Gráfico con el porcentaje de acierto en el idioma de los tweets*

Al revisar el idioma, el número de tweets con texto en español es 12, en inglés 7 y en catalán 1. Vemos que desaparece el euskera, dado que está mal etiquetado.

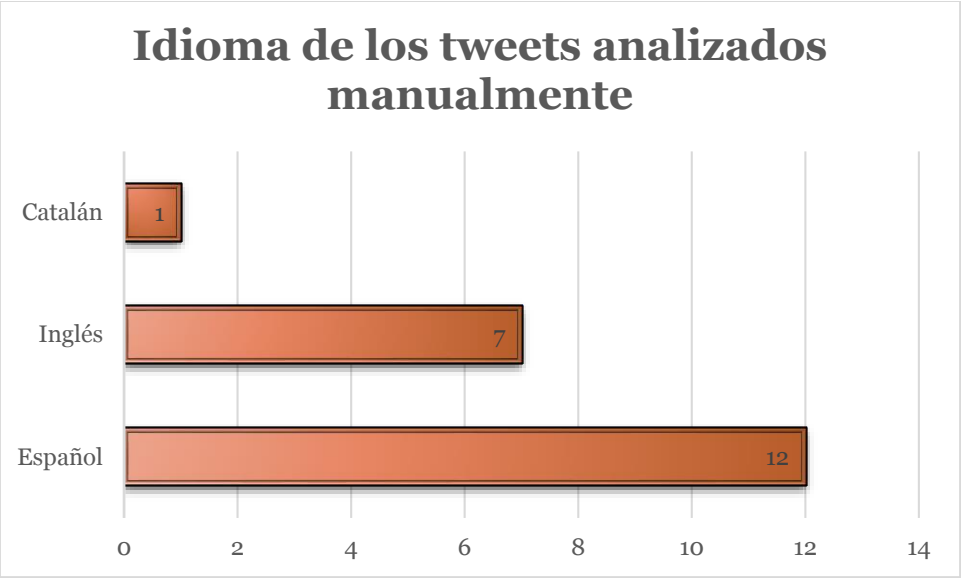


*Ilustración 25. Idioma de los tweets analizados manualmente*

# 7. Conclusión

La recopilación de información de redes sociales proporciona una fuente de datos con gran proyección a la hora de su análisis. El procesado de lenguaje natural y la geolocalización de los tweets han sido los puntos más importantes de la aplicación.

Si recordamos los objetivos del proyecto, vemos que se ha podido recopilar los tweets a través del API que proporciona Twitter, pudiendo tanto mostrar los tweets por terminal así como guardar los datos en csv. La dependencia de los resultados a enfocarse al timeline de un usuario en concreto, así como la probable repetición de tweets debido al tiempo entre llamadas, puede ser algo negativo en casos concretos del uso de la aplicación.

La clasificación de los tweets según su zona geográfica ha estado fuertemente condicionada por la escasez de tweets que incluyan su geolocalización. La solución a este problema ha llevado a la obtención del lugar del usuario que aparece en el perfil, con la pérdida de exactitud que esto supone. No obstante, dado que muchos usuarios no indican un lugar válido tampoco en su perfil, el número de tweets geolocalizados en territorio nacional sigue siendo pequeño.

El análisis lingüístico de los textos de Twitter de forma automática se ha basado en la librería NLTK. Pero en las stopwords que proporciona este módulo no figuraban las lenguas autonómicas, por tanto, la búsqueda de estas palabras vacías en otros idiomas ha supuesto revisar alternativas para su obtención, y se han tenido problemas con la codificación de estos archivos y el estándar del módulo.

En cuanto a proporcionar una visualización significativa de los datos obtenidos, si bien el mapa cumple con sus objetivos de marcar la localización de tweets, seguir un paso más allá y hacer la web más interactiva, hubiera facilitado de cara al usuario una mayor comprensión de la aplicación y su reutilización de formas diversas.

Los principales problemas que me he enfrentado al embarcarme con este proyecto, ha sido mi poca formación en cuanto a las herramientas que me había planteado utilizar. Puesto que no había utilizado el API de Twitter, ni había trabajado con Python hasta este último curso del master, y nunca había realizado ninguna página web con PHP. Pese a ello, con el debido tiempo de estudio y formación, puedo decir que el resultado del proyecto en general ha sido satisfactorio. Sin embargo, se podría haber mejorado substancialmente la aplicación así como reducido el tiempo de este trabajo de haber estado más versada en la materia.

En cuanto a posible desarrollo de este trabajo, me hubiera gustado poder proporcionar una capa de vista al usuario para facilitar la utilización de la aplicación. También mejorar la visualización de los tweets tanto a nivel de mapas, por ejemplo mapas de calor, como a nivel de mejora de análisis de datos. Serían estos puntos los que podrían dar pie a mejoras futuras.

En mi opinión, sus posibles aplicaciones en el campo del estudio de los perfiles de usuario e idioma son muy significativos. Otras posibles situaciones de estudio serían los flujos turísticos por un territorio en concreto, el avance lingüístico de un lenguaje o la segmentación de la publicidad por geografía e idioma.

# 8. Referencias

[1] Twitter as a Corpus for Sentiment Analysis and Opinion Mining (2010) Alexander Pak, Patrick Paroubek

[2] Tweepsmap http://tweepsmap.com/

[3] Trendsmap http://trendsmap.com/

[4] Twaps http://joshkrajnak.com/twaps/

[5] The one million tweet map http://onemilliontweetmap.com/

[6] Formato JSON https://es.wikipedia.org/wiki/JSON

[7] Twitter y OAuth https://dev.twitter.com/oauth

[8] OAuth http://oauth.net/

[9] Página oficial de Tweepy http://www.tweepy.org/

[10] Librería Open Source Twitter API https://github.com/geduldig/TwitterAPI

[11] Librería Open Source Python-Twitter https://github.com/bear/python-twitter

[12] K. Leetaru, S. Wang, G. Cao, A. Padmanabhan, and E. Shook, "Mapping the global Twitter heartbeat: The geography of Twitter," First Monday, vol. 18, no. 5, abril de 2013. Disponible: http://firstmonday.org/ojs/index.php/fm/article/view/4366 [Accedido: 10 de marzo de 2015]

[13] Listado Excel elaborado a partir de datos del INE y Google Maps de htttp://www.businessintelligence.info

[14] Xlrd https://pypi.python.org/pypi/xlrd

[15] Documentación librería JSON http://docs.python-guide.org/en/latest/scenarios/json/

[16] Detección idioma en Python con NLTK http://programacion-mas.blogspot.com.es/2013/07/detectar-idioma-python-nltk.html

[17] Stopwords http://www.ranks.nl/stopwords/galician

[18] Tweepy http://stackoverflow.com/questions/8137777/tweepy-entities-appengine

[19] CSV https://es.wikipedia.org/wiki/CSV

[20] Leaflet http://leafletjs.com/

Diferencias entre API Rest y Streaming de Twitter https://dev.twitter.com/streaming/overview

Librería NLTK https://es.wikipedia.org/wiki/NLTK http://www.nltk.org/index.html

Glosario de términos: http://www.cosmociudadano.mx/glosario-de-terminos-para-twitter/

Stopwords

https://github.com/iucl/l2-writing-assistant/tree/master/babelnet-api-2.0/resources/jlt/stopwords

https://wiki.apache.org/solr/LanguageAnalysis

Artículo con ejemplos de trabajos de geolocalización:
http://www.hablandoencorto.com/2013/12/herramientas-twitter-geolocalizacion.html

# 9. Bibliografía

Matthew A. Russell (2013), Mining the Social Web Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub. 2ª Edición. O'Reilly Media Inc.

Bird, Steven, Edward Loper and Ewan Klein (2009), Natural Language Processing with Python. O'Reilly Media Inc.

José Hernández Orallo, M.José Ramírez Quintana, Cèsar Ferri Ramírez (2004), Introducción a la minería de datos. Editorial Pearson

Temario de la asignatura Explotación de Datos Masivos con D. César Ferri Ramírez

# 10. Apéndice 1: Lista de Tweets

TWEET Num.1
Texto mensaje: #GanasDe salir.
Idioma del tweet: basque
Lugar del tweet: None
Lugar del usuario: Entre Hyrule y Termina
Coordenadas del usuario: Lugar no identificado

TWEET Num.2
Texto mensaje: The Infection - Disturbed ♫ http://t.co/YdLOveOEne
Idioma del tweet: english
Lugar del tweet: None
Lugar del usuario: Entre Hyrule y Termina
Coordenadas del usuario: Lugar no identificado

TWEET Num.3
Texto mensaje: "Data does not change the world, but #opendata makes the change possible"- President Kikwete #africaopendata @Data4Africa vía @epsiplatform
Idioma del tweet: english
Lugar del tweet: None
Lugar del usuario: España
Coordenadas del usuario: Lugar no identificado

TWEET Num.4
Texto mensaje: RT @xataka: No, los precios de Netflix en España todavía no son oficiales, pero esto es lo que espera... http://t.co/8m5glTRkAM http://t.co…
Idioma del tweet: spanish
Lugar del tweet: None
Lugar del usuario:
Coordenadas del usuario: Lugar no identificado

TWEET Num.5
Texto mensaje: Gira, el mundo gira... y nuestras sillas también. ¡Sólo hasta el 13 de septiembre! http://t.co/l0X6egXZ5s http://t.co/k1J7QNf9Kp
Idioma del tweet: spanish
Lugar del tweet: None
Lugar del usuario: Spain
Coordenadas del usuario: Lugar no identificado

TWEET Num.6
Texto mensaje: what https://t.co/lZNALInd0W
Idioma del tweet: english

Lugar del tweet: None
Lugar del usuario: Valencia, España
Coordenadas del usuario: 39.47024 , -0.3768049

TWEET Num.7
Texto mensaje: Senior Java Test Automation Engineer Dell Secureworks @Dell #Edinburgh http://t.co/mckwdIAh7b #compilers #Javacode #teststrategy
Idioma del tweet: basque
Lugar del tweet: None
Lugar del usuario: Dublin
Coordenadas del usuario: Lugar no identificado

TWEET Num.8
Texto mensaje: Structured vs. unstructured - which data is more important? http://t.co/JAqKQ6apsS
Idioma del tweet: english
Lugar del tweet: None
Lugar del usuario: Seattle, WA
Coordenadas del usuario: Lugar no identificado

TWEET Num.9
Texto mensaje: Emmys 2015: Mejor actor y actriz de comedia http://t.co/wpWEk5gg4D http://t.co/ynY9EvlDDZ
Idioma del tweet: english
Lugar del tweet: None
Lugar del usuario:
Coordenadas del usuario: Lugar no identificado

TWEET Num.10
Texto mensaje: Sailor Moon vuelve a Super 3 http://t.co/Ba6CjQcPtg #rp2 #manga #anime
Idioma del tweet: catala
Lugar del tweet: None
Lugar del usuario: España
Coordenadas del usuario: Lugar no identificado

TWEET Num.11
Texto mensaje: RT @Ele_a_secas: ROCK, CERVEZA Y PATATAS BRAVAS, EL MEJOR INVENTO @MdT_TVE @olivares_javier @Abixaf @MdMinistericos @rodolferasfans http://…
Idioma del tweet: english
Lugar del tweet: None
Lugar del usuario: 40.411792, -3.706802
Coordenadas del usuario: Lugar no identificado

TWEET Num.12
Texto mensaje: Embracing The Torture - Devil You Know ♫ http://t.co/433Z5tgOct

Idioma del tweet: english
Lugar del tweet: None
Lugar del usuario: Entre Hyrule y Termina
Coordenadas del usuario: Lugar no identificado

TWEET Num.13
Texto mensaje: Consulta los grados de la UPV con plazas libres para el curso 2015-2016: http://t.co/ntAWWkfdB4
Idioma del tweet: spanish
Lugar del tweet: None
Lugar del usuario: Alcoy, Gandia, Valencia
Coordenadas del usuario: 38.69763, -0.4730959

TWEET Num.14
Texto mensaje: Hasta el día 11, último plazo para preinscribirse en los másteres universitarios UPV http://t.co/kj5FuQlveA (listado de plazas libres)
Idioma del tweet: spanish
Lugar del tweet: None
Lugar del usuario: Alcoy, Gandia, Valencia
Coordenadas del usuario: 38.69763, -0.4730959

TWEET Num.15
Texto mensaje: La segunda reseña sobre la novela de G.P. Danilevsky nos llega de la mano de Daniel Blanco en su blog «La ventana... http://t.co/fNm8Y0VRUw
Idioma del tweet: spanish
Lugar del tweet: None
Lugar del usuario: Asturias
Coordenadas del usuario: 43.26667, -6.616667

TWEET Num.16
Texto mensaje: Nomadic - Slipknot ♫ http://t.co/hlY7RGYCGd
Idioma del tweet: basque
Lugar del tweet: None
Lugar del usuario: Entre Hyrule y Termina
Coordenadas del usuario: Lugar no identificado

TWEET Num.17
Texto mensaje: Hoy compartimos con vosotros dos reseñas de "La princesa Tarakanova". La primera de ellas nos llega de la mano de... http://t.co/szSE6A9eLY
Idioma del tweet: spanish
Lugar del tweet: None
Lugar del usuario: Asturias
Coordenadas del usuario: 43.26667, -6.616667

TWEET Num.18

Texto mensaje: Es muy interesante el papel de @equo y su capacidad de integrarse en coaliciones. https://t.co/8yJo0NFyvA
Idioma del tweet: spanish
Lugar del tweet: None
Lugar del usuario: Valencia
Coordenadas del usuario: 39.47024, -0.3768049

TWEET Num.19
Texto mensaje: Oferta del #SERVEF de #tècnic elec. (automòbil) #Massanassa (439729).Inscripcions: http://t.co/nM7mQMNbrc http://t.co/9tcbczDUzs
Idioma del tweet: catala
Lugar del tweet: None
Lugar del usuario: Valencia (España)
Coordenadas del usuario: 39.47024, -0.3768049

TWEET Num.20
Texto mensaje: El templo de las linternas, el Okunoin.

Flickr: Miquel Lleixà Mora. https://t.co/NmhK8b2vwZ http://t.co/wGbkQaOTbC
Idioma del tweet: spanish
Lugar del tweet: None
Lugar del usuario: Tokyo
Coordenadas del usuario: Lugar no identificado

# 11. Apéndice 2: Código de la aplicación

- Código de la aplicación de Python, del archivo llamado `twit.py`

```
#!/usr/bin/env python
import tweepy
import json
from xlrd import open_workbook
import xlrd
#Import the necessary methods from tweepy library
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream
from HTMLParser import HTMLParser
from TwitterAPI import TwitterAPI
import nltk

#Variables que contienen las credenciales de usuario para
acceder al Twitter API
access_token = "XXXXXXXXXXXXXXXXXXXXXXXX"
access_token_secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
consumer_key = "XXXXXXXXXXXXXXXXXXXXXXXX"
consumer_secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"

def on_data(self, data):
        data_now = HTMLParser().unescape()
        data = json.loads(data_now)
        tweet = data['text']
        print tweet
        return True
#This is a basic listener that just prints received tweets to
stdout.
class StdOutListener(StreamListener):

    def on_status(self, status):
        print status.TEXT
        if status.coordinates:
            print 'coords:', status.coordinates
        if status.place:
            print 'place:', status.place.full_name
        return True

    on_event=on_status

    def on_error(self, status):
        print status

def placeSpain(latitud, longitud):
```

```
    file=open_workbook('listalongitudlatitud.xls')
    sheet = file.sheet_by_index(0)
    rango = False
    for indice in range(1, sheet.nrows):
        line = sheet.row(indice)
        if latitud == line[3].value and longitud
==line[4].value:
            rango = True
            break
    return rango

def idiomaTexto(texto):

    #Lista de idiomas disponibles en la nltk
    languages=["spanish","english","catala", "basque",
"galician"]
    #Texto a analizar
    text_to_detect = texto
    #Dividimos el texto de entrada en tokens o palabras unicas
    tokens = nltk.tokenize.word_tokenize(text_to_detect)
    tokens = [t.strip().lower() for t in tokens] # Convierte
todos los textos a minusculas para su posterior comparacion
    #Creamos un dict donde almacenaremos la cuenta de las
stopwords para cada idioma
    lang_count = {}
    #Por cada idioma
    for lang in languages:
        #Obtenemos las stopwords del idioma del modulo nltk
     stop_words = unicode(nltk.corpus.stopwords.words(lang))
     lang_count[lang] = 0 #Inicializa a 0 el contador para cada
idioma
     #Recorremos las palabras del texto a analizar
     for word in tokens:
          if word in stop_words: #Si la palabra se encuentra
entre las stopwords, incrementa el contador
               lang_count[lang] += 1
    #Obtiene el idioma con el numero mayor de coincidencias
    detected_language = max(lang_count, key=lang_count.get)
    return detected_language

def placeUsuario(municipio):
    file=open_workbook('listalongitudlatitud.xls')
    sheet = file.sheet_by_index(0)
    encontrado = 'Lugar no identificado'
    array_separadores = [", ", " "]
    listamunicipio = ""
    for separador in array_separadores:
        listamunicipio=municipio.split(separador)
        if len(listamunicipio) > 2:
            break
    for indice in range(1, sheet.nrows):
        line = sheet.row(indice)
        #1 provincia, 2 poblacion en excel ej valencia, espanya
        #if listamunicipio[1] == line[1].value or
listamunicipio[1] == line[2].value or listamunicipio[2]==
line[1].value:
```

```
        if len(listamunicipio) > 0 and (listamunicipio[0] ==
line[1].value or listamunicipio[0] == line[2].value):
            encontrado=str(line[3].value)+",
"+str(line[4].value)
            break
    return encontrado

if __name__ == '__main__':

    #api = TwitterAPI(consumer_key, consumer_secret,
access_token, access_token_secret)
    #This handles Twitter authetification and the connection to
Twitter Streaming API
    #l = StdOutListener()
    auth = OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    #on_data(1, '{"text":"&quot;"}')
    api = tweepy.API(auth)

    f=open('geotuits.csv', 'w')
    idioma = "Desconocido"
    public_tweets = api.home_timeline()
    cont=1
    for tweets in public_tweets:
        print "\nTWEET Num."+str(cont)
        print "Texto mensaje: "+tweets.text #Texto prueba
        idioma = idiomaTexto(tweets.text)
        f.write("TWEET Num"+str(cont)+" ")
        f.write(idioma+'\r\n') #Idioma del texto de prueba
        print "Idioma del tweet: "+idioma
        print "Lugar del tweet: "+str(tweets.place) #Lugar del
tweet si lo tiene, sino 'None'
        print "Lugar del usuario: "+tweets.author.location
        if tweets.coordinates and
placeSpain(tweets.coordinates[0][0][0][0],
tweets.coordinates[0][0][0][1]):
            f.write(tweets.coordinates)
            print "Coordenadas del tweet: "+tweets.coordinates
        #if tweets.author.location is not "":
        coordUsu = placeUsuario(tweets.author.location)
        print "Coordenadas del usuario: "+str(coordUsu)
        f.write(str(coordUsu))
        cont=cont+1
    print "fin"
```

- Código de la web en PHP, `geoWebTwit.php`
    - Con la versión 0.7.4 de Leaflet

```
<!DOCTYPE html><html>
<meta charset="utf-8" />
<head>
     <script src="http://cdn.leafletjs.com/leaflet-
0.5/leaflet.js"></script>
     <link rel="stylesheet"
href="http://cdn.leafletjs.com/leaflet-0.5/leaflet.css" />

 <style>
  #map {
  widh: 50px;
  height: 800px; }
 </style>

 </head>
  <body>
   <div id="map"></div>
<?php
$registros = array();
$fichero = fopen("geotuits.csv", "r");
    // Lee los registros
    while ($datos = fgetcsv($fichero, 0, ",", "\"", "\"")) {
        // AÃ±ade el registro leido al array de registros
        $registros[] = $registro;
    }

    fclose($fichero);

    echo "Leidos " . count($registros) . " tweets\n";
     $coordenadas_now = "";
    for ($i = 0; $i < count($registros); $i++) {
        echo "Coordenadas Tweet: " . $registros[$i]["Idioma"] .
"\n";
     $coordenadas_now .=
"L.marker([".$registros[$i]["Idioma"]."], {draggable:
true}).addTo(map);";
    }
?>

<script>
var map = L.map('map').
setView([41.66, -4.72], 6);

L.tileLayer('http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png',
{
    attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors,
<a href="http://creativecommons.org/licenses/by-sa/2.0/">CC-BY-
SA</a>, Imagery Â© <a href="http://cloudmade.com">CloudMade</a>
Modificado por Elena Asensio',
    maxZoom: 18
```

```
}).addTo(map);


L.control.scale().addTo(map);
<?php echo $coordenadas_now; ?>
 </script>

 </body>
 </html>
```

- Con la versión 0.7.5 de Leaflet

```
<!DOCTYPE html><html>
<meta charset="utf-8" />
<head>
     <script
src="http://leafletjs.com/dist/leaflet.js"></script>
     <link rel="stylesheet"
href="http://leafletjs.com/dist/leaflet.css" />

 <style>
  #map {
  widh: 50px;
  height: 800px; }
 </style>

 </head>
  <body>
   <div id="map"></div>
<?php
$registros = array();
$fichero = fopen("geotuits.csv", "r");
    // Lee los registros
    while ($datos = fgetcsv($fichero, 0, ",", "\"", "\"")) {
        // AÃ±ade el registro leido al array de registros
        $registros[] = $registro;
    }

    fclose($fichero);

    echo "Leidos " . count($registros) . " tweets\n";
      $coordenadas_now = "";
    for ($i = 0; $i < count($registros); $i++) {
        echo "Coordenadas Tweet: " . $registros[$i]["Idioma"] .
"\n";
     $coordenadas_now .=
"L.marker([".$registros[$i]["Idioma"]."], {draggable:
true}).addTo(map);";
    }
?>

<script>
var map = L.map('map').
setView([41.66, -4.72], 6);
```

```
L.tileLayer('https://api.tiles.mapbox.com/v4/{id}/{z}/{x}/{y}.pn
g?access_token=pk.eyJ1IjoibWFwYm94IiwiYSI6IjZjNmRjNzk3ZmE2MTcwOT
EwMGY0MzU3YjUzOWFmNWZhIn0.Y8bhBaUMqFiPrDRW9hieoQ', {
      maxZoom: 18,
      attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors,
' +
           '<a href="http://creativecommons.org/licenses/by-
sa/2.0/">CC-BY-SA</a>, ' +
           'Imagery Â© <a href="http://mapbox.com">Mapbox</a>',
      id: 'mapbox.streets'
}).addTo(map);

<?php echo $coordenadas_now; ?>
 </script>

 </body>
 </html>
```

# 11. Apéndice 3: Utilización de la aplicación

En primer lugar, se debe indicar los tokens de seguridad de la cuenta de Twitter que vamos a vincular. Para ello, abriremos el archivo con el código de la aplicación, llamado:

`twit.py`

Ahora cambiaremos el valor entre comillas de las variables:

```
access_token = "XXXXXXXXXXXXXXXXXXXXXXXXX"
access_token_secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
consumer_key = "XXXXXXXXXXXXXXXXXXXXXXXXX"
consumer_secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
```

Guardamos el archivo y desde nuestra terminal indicamos:

```
sudo chmod a+x twit.py
```

Ahora al abrir el archivo preguntará si deseas ejecutar el programa en la terminal, y ya estará listo.

Cuando acabe la ejecución podemos ir a nuestra página web desde el navegador y ver el mapa generado:

```
localhost/geoWebTwit.php
```

Las restricciones de Twitter en las cuentas, hacen que sólo podamos obtener como máximo una lista de 300 tweets cada 15 minutos, para tener ese máximo de tweets hemos modificado el código en el `main` para que recoja todos los datos durante 5 minutos tal y como aparece a continuación:

```
if __name__ == '__main__':

    #api = TwitterAPI(consumer_key, consumer_secret,
access_token, access_token_secret)
    #This handles Twitter authetification and the connection to
Twitter Streaming API
    #l = StdOutListener()
    auth = OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    #on_data(1, '{"text":"&quot;"}')
    api = tweepy.API(auth)
```

```
    f=open('geotuits.csv', 'w')
    idioma = "Desconocido"
    cont=1
    timeout=time.time()+60*5#5 minutos
    while True:
        test=0
        public_tweets = api.home_timeline()
        for tweets in public_tweets:
            print "\nTWEET Num."+str(cont)
            print "Texto mensaje: "+tweets.text #Texto prueba
            idioma = idiomaTexto(tweets.text)
            f.write("TWEET Num"+str(cont)+" ")
            f.write(idioma+'\r\n') #Idioma del texto de prueba
            print "Idioma del tweet: "+idioma
            print "Lugar del tweet: "+str(tweets.place) #Lugar
del tweet si lo tiene, sino 'None'
            print "Lugar del usuario: "+tweets.author.location
            if tweets.coordinates and
placeSpain(tweets.coordinates[0][0][0][0],
tweets.coordinates[0][0][0][1]):
                f.write(tweets.coordinates)
                print "Coordenadas del tweet:
"+tweets.coordinates
            #if tweets.author.location is not "":
            coordUsu = placeUsuario(tweets.author.location)
            print "Coordenadas del usuario: "+str(coordUsu)
            f.write(str(coordUsu))
            cont=cont+1
        if test == 5 or time.time()>timeout:
            break
        test=test-1
        time.sleep(1)
    print "fin"
```

El gran inconveniente es que es probable que un gran número de esos 300 tweets sean repetidos. Por ejemplo en nuestro mapa aparecen pocos sitios localizados porque son repetidos o el municipio es el mismo.

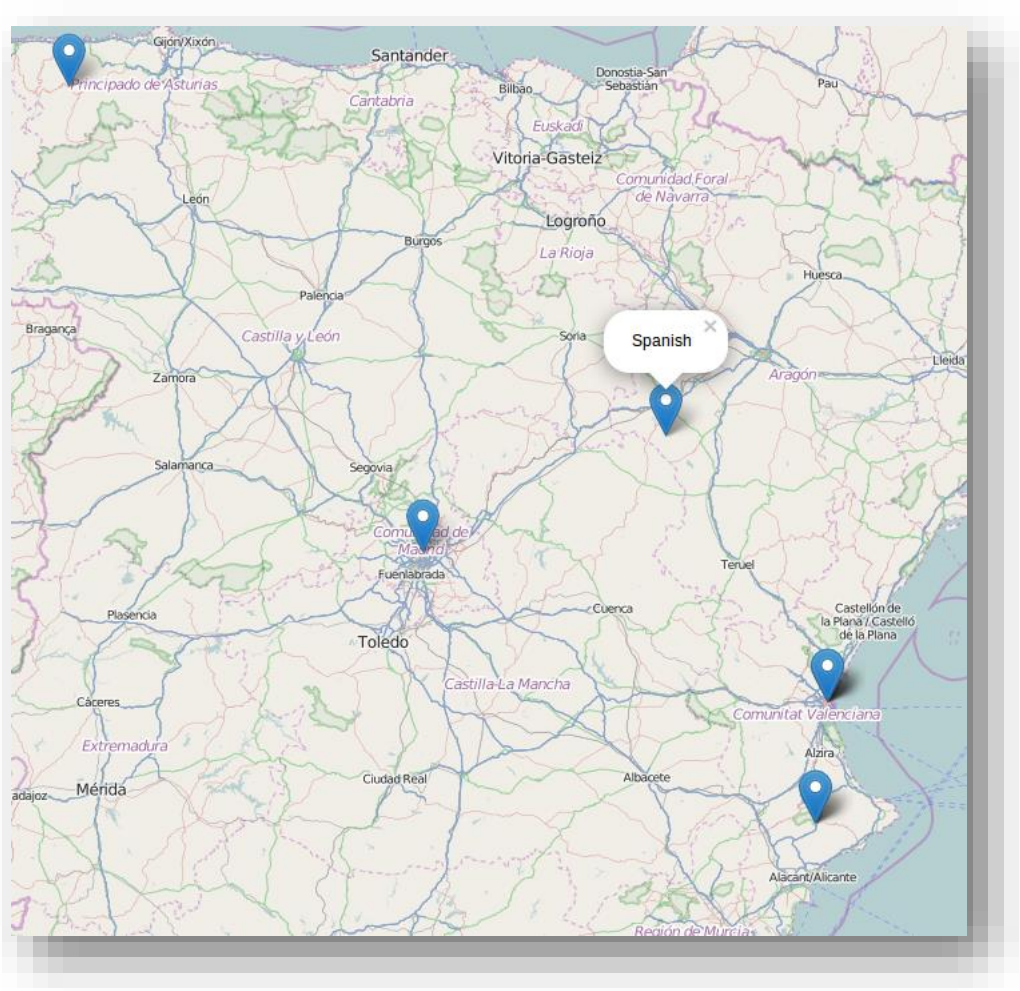


*Ilustración 26. Captura del mapa web con muestra de 300 tweets*

Una opción válida es hacer que el programa se ejecute cada 15 minutos hasta llegar al tamaño de muestra deseado, y así recogería, por ejemplo, en una hora 1200 tweets, como vemos en la captura del mapa de la web a continuación.


*Ilustración 27. Captura del mapa de la web con una muestra de 1200 tweets*