Universidad de La Laguna

Trabajo de Fin de Máster

Predicción de resultados en procesos electorales usando una herramienta de análisis de redes sociales

Predicting results on electoral process using a social network analysis tool

Adrián Rodríguez Vargas

San Cristóbal de La Laguna, 2 de Febrero de 2018

D. Jesús Miguel Torres Jorge, con DNI número 43.826.207-Y, profesor Contratado Doctor adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutor

CERTIFICA

Que la presente memoria titulada:

“Predicción de resultados en procesos electorales usando una herramienta de análisis de redes sociales”

ha sido realizada bajo su dirección por D. Adrián Rodríguez Vargas con DNI número 78.698.319-R.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en San Cristóbal de La Laguna a 5 de febrero de 2018

Agradecimientos

A mis padres, Teresa y Jesús por haberme dado tanto en la vida, y a mi hermano Jesús, por estar siempre ahí.

A Alberto Miguel Rodríguez Orihuela, por todas las cosas que aprendimos juntos con Smetrica.com y sin el que nunca hubiera sido posible que dicha herramienta existiera.

A Jesús Miguel Torres Jorge, mi tutor, por dirigir este proyecto a más de 3.000 km de distancia.

A los amigos que me han apoyado para terminar aquellos proyectos que permanecían abiertos.

Licencia

Resumen

Durante el año 2012 participé, junto con Alberto M. Rodríguez Orihuela, en la creación de smetrica.com, una herramienta para analizar crecimiento e interacción en *fanpages* de Facebook. Con ella obtuvimos el primer puesto en los premios PCTT Emprende.ull[1].

A lo largo de la vida del proyecto hubo un amplio grupo de personas que usaban la herramienta como apoyo de estudios de ciencias sociales, y en muchos casos orientados a la política. Tal fue así que en repetidas ocasiones realizamos la extracción de datos bajo demanda para ciertas personas y organizaciones. Estos datos contenían información más detallada que el proyecto inicial no contemplaba, incluyendo contenido de la publicación así como comentarios generados por usuarios de la plataforma Facebook.

Dando un paso más allá en este trabajo de Fin de Máster se ha realizado una nueva Herramienta de análisis de redes sociales que pueda servir para la predicción de resultados de procesos electorales, tomando como entrada ciertos parámetros como número de usuarios, interacción, contenido de los comentarios realizados por los usuarios, etc.

Palabras clave: redes sociales, Facebook, API, predicción, elecciones, política, sociología.

[1] Smetrica, un proyecto de medición en redes sociales, obtiene el primer puesto de los premios PCTT Emprende.ull 2012: http://t.ull.es/6zp

Abstract

During 2012, me and Alberto M. Rodríguez Orihuela, developed smetrica.com, a tool for analyzing growth and interaction on Facebook fanpages. With this tool we won the first award on PCTT Emprende.ull[2].

A group of people were using the tool and the data for backing researches about social science and more specifically backing research about politics. In fact, several times we did some custom data extraction for some of these people and organizations. The extracted data contained more detailed information that initial project had, as the post content and also comments written by the users of Facebook platform.

Going a step forward, during the fulfillment of this Master's Thesis, a new tool has been developed: Social Network Analysis tool able to predict electoral processes results, having as an input parameter such number of users, interaction, comment written by each user, etc.

Keywords: social network, Facebook, API, prediction, electoral process, politics, sociology

[2] Smetrica, measurement tool for social networks, winner of PCTT Emprende.ull 2012 award: http://t.ull.es/6zp (Spanish)

Índice general

1. Introducción

Fue durante las elecciones presidenciales de Estados Unidos de 2008 cuando las redes sociales comenzaron a ser usadas por primera vez de forma intensiva, especialmente por el partido Demócrata de Obama, durante la etapa de recaudación de fondos. Y posteriormente con el comienzo de la campaña donde los electores participaron en la contienda en primera persona con voz propia, apoyando o criticando a cada partido político.

A partir de entonces todas las organizaciones políticas de los países desarrollados aumentaron sus presupuestos para estrategias en redes sociales en las posteriores campañas electorales.

Sin embargo, a pesar de tener un amplio aumento del uso de las redes sociales tanto por parte de las organizaciones políticas, como por parte de los electores durante las campañas electorales, sigue existiendo un gran margen de error entre las predicciones sobre intención de voto y la realidad. Así como entre las encuestas y sondeos y los resultados finales cuando finaliza la jornada electoral.

1.1. Objetivos

Con este trabajo se pretende tener un herramienta fiable que sirva para predecir el resultado de un proceso electoral, tomando como entrada datos de cada fanpage de Facebook, extrayendo cada una de las publicaciones que los usuarios o la propia fanpage realiza, así como los likes como los comentarios que realizan en las mismas, así como también análisis del sentimiento de dichas publicaciones y comentarios.

La herramienta debería ser capaz, además, de complementar los Barómetros del CIS de enero, abril, julio y octubre en los cuales se incluyen preguntas fijas sobre actitudes políticas a partir de las que el CIS calcula y publica la estimación de voto [3].

[3] Fuente: Centro de Investigaciones Sociológicas: Barómetros. http://www.cis.es/cis/opencm/ES/11_barometros/index.jsp

De esta forma se pretende tener una muestra mucho mayor de la que se toma en el CIS, cuyo tamaño de la muestra de 2.500 personas[4], con esta herramienta se tendrá una muestra limitada sólo por el número de usuarios de una fanpage de Facebook que interaccionan en la misma.

Otro objetivo de esta herramienta es la de ser capaz de evaluar el grado de satisfacción general de los usuarios que participan activamente con la fanpage de cada partido político o la fanpage de un candidato, escribiendo publicaciones o comentando las mismas.

1.2. Motivaciones

El principal aliciente de llevar a cabo este proyecto es el hacer algo de relevancia con datos que están disponibles para quien quiera usarlos de forma pública, gracias a la API[5] de Facebook, sin necesidad de hacer ningún esfuerzo en recogida de datos a pie de campo de forma individual. Especialmente en un tema tan relevante como la política, que determina la vida de tantas personas en el día a día.

Otra motivación es la de usar herramientas de análisis de lenguaje natural, así como otras utilidades de machine learning.

[4] Nota de investigación sobre la metodología general de los barómetros mensuales del Centro de Investigaciones Sociológicas.
http://www.cis.es/cis/export/sites/default/-Archivos/NotasdeInvestigacion/NI004_MetodologiaBarometros_Informe.pdf

[5] Application Programming Interface o Interfaz de programación de aplicaciones, conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.
https://es.wikipedia.org/wiki/Interfaz_de_programaci%C3%B3n_de_aplicaciones

2. Estado del arte

En la actualidad, el único indicador público de estimación de los resultados electorales es realizado por el Centro de Investigaciones Sociológicas, el CIS, que depende del Ministerio de la Presidencia y para las Administraciones Territoriales.

Los barómetros se realizan con una periodicidad mensual –excepto los meses de agosto– y tienen como principal objetivo medir el estado de la opinión pública española del momento. Para ello se entrevista en torno a 2.500 personas elegidas al azar dentro del territorio nacional, de las que, además de sus opiniones, se recoge una amplia información social y demográfica para el análisis.

Únicamente, durante los meses de enero, abril, julio y octubre, así como en las encuestas pre-electorales, los barómetros incluyen un conjunto de preguntas fijas sobre actitudes políticas a partir de las que el CIS calcula y publica la estimación de voto. Los puntos que cubre estos barómetros trimestrales son:

- Valoración de la actuación del gobierno
- Valoración de la actuación de la oposición
- Confianza en el presidente del gobierno
- Confianza en el líder de la oposición
- Valoración de los líderes de los principales partidos políticos
- Valoración de los miembros del gobierno
- Valoración de la situación económica dentro de un año con respecto a la actual
- Valoración de la situación política dentro de un año con respecto a la actual
- Intención de voto en las elecciones generales
- Simpatía por los partidos políticos

Los resultados de los barómetros mensuales se hacen públicos a través de la web del CIS inicialmente en formato de “avance de resultados”. Tras la finalización del resto de procesos técnicos, incluida la anonimización, los datos del estudio pasan a formar parte del Banco de Datos del CIS, momento en el que quedan disponibles en

la página web el fichero de microdatos del estudio y el resto de la documentación asociada.

2.1. Estimación de la intención de voto

A partir de la intención de voto declarada por las personas entrevistadas, y de otros datos de la encuesta, se aplica un modelo estadístico para realizar la estimación del voto. El CIS proporciona tanto los resultados de las respuestas a la pregunta de intención de voto como los de la aplicación del modelo de estimación. Otros modelos aplicados a los mismos datos pueden dar lugar a estimaciones de los resultados electorales diferentes.

2.2. Presentación de datos y comparativas

A continuación, se presentan en forma de tabla la estimación de voto extraída de la encuesta preelectoral del CIS de mayo de 2016 en su informe de avance de resultados, así como también se presentan los datos del escrutinio, en porcentajes, de las elecciones generales de junio de 2016.

Por último en la página 15 se muestra un gráfico con la comparativa entre la estimación de voto de estimada por el CIS y los resultados de las elecciones generales.

Tabla de datos sobre intención de voto en la encuesta preelectoral del CIS de mayo de 2016, número 3141:

	Voto directo en la encuesta (en %)	Estimación de voto CIS (en % sobre voto válido)	Estimación de escaños
PP [6]	16.8	29.2	118-121
PSOE [7]	14.6	21.2	78-80
Unidos Podemos [8]	11.8	16.4	56-60
Ciudadanos	8.5	14.6	38-39
En Comú Podem	3.3	4.5	14.15
Compromís-Podemos-EUPV	1.4	3.0	9-10
ERC	1.8	2.4	8-9
CDC	1.0	1.8	6-7
En Marea	1.7	1.7	7
PNV	0.8	1.2	6
EH Bildu	0.5	0.9	3
CC	0.1	0.2	-
Otros partidos	1.2	2.1	
En blanco	2.7	0.7	
Voto nulo	0.7		
Abstención (“No votaría”)	11.0		
No sabe	14.7		
No contesta	7.3		

Fuente: Barómetro de mayo de 2016. Estudio número 3141
http://datos.cis.es/pdf/Es3141mar_A.pdf

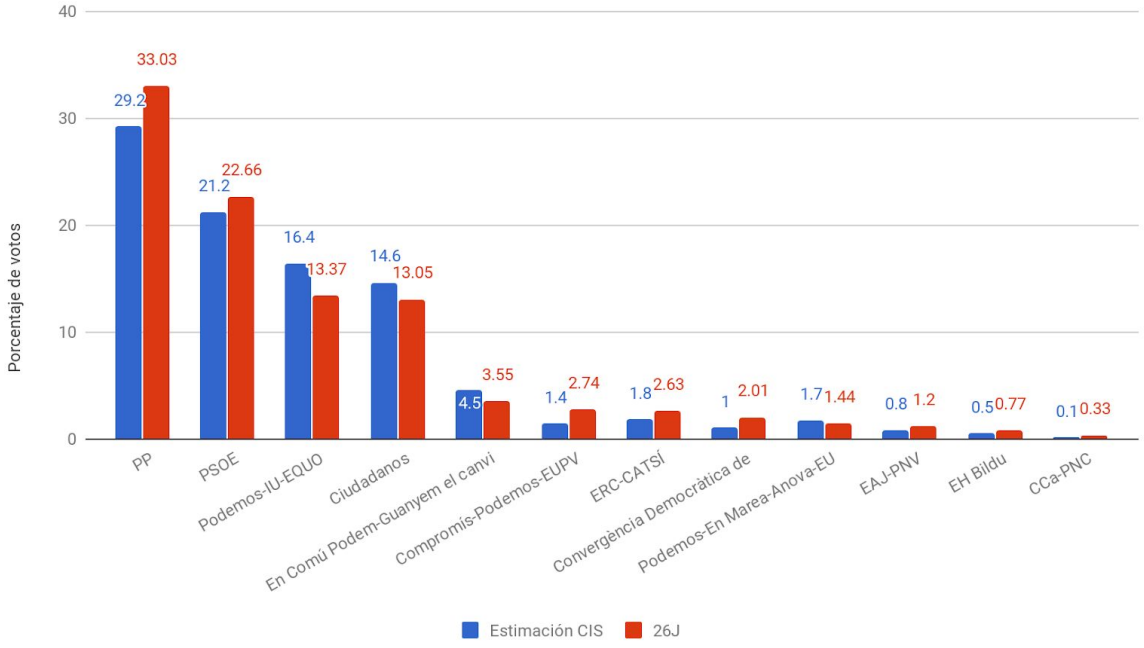
[6] Incluye los resultados de las coaliciones con UPN (en Navarra), con PAR (en Aragón) y con Foro (en Asturias)
[7] Incluye el resultado de la coalición con Nueva Canarias
[8] Incluye los resultados de la coalición con MÉS (en Baleares)

A continuación se muestra el porcentaje de votos obtenidos durante las elecciones generales del 26 de junio de 2016 por los partidos que obtuvieron representación.

Partido político	Porcentaje de votos
PP	33.03
PSOE	22.66
Podemos-IU-EQUO	13.37
Ciudadanos	13.05
En Comú Podem-Guanyem el canvi	3.55
Compromís-Podemos-EUPV	2.74
ERC-CATSÍ	2.63
Convergència Democràtica de Catalunya	2.01
Podemos-En Marea-Anova-EU	1.44
EAJ-PNV	1.20
EH Bildu	0.77
CCa-PNC	0.33

Fuente: Ministerio del Interior, Gobierno de España
http://resultados2016.infoelecciones.es/99CO/DCO99999TO.htm

Se observa la comparativa entre la encuesta preelectoral del CIS y resultados en las elecciones generales del 26 de Junio de 2016



Una vez comparados los datos reales de la estimación de voto calculada por el CIS, podemos observar que los barómetros muestran una estimación bastante fiable.

3. Diseño

La idea principal se centra en la predicción de resultados en un proceso electoral determinado basado en datos procedentes de la red social Facebook, para ser concretos de su API. Para ello se pretende desarrollar una aplicación usando el framework de Python, Django en su versión 1.11 (última versión LTS[9] disponible). He escogido esta tecnología pues trabajo con ella a diario y no requiere tiempo extra de aprendizaje.

3.1. Estructura de datos en Facebook

Antes de comenzar con los requisitos de software, es necesario hacer una breve introducción sobre como Facebook organiza los datos que van a ser usados aquí.

Página de Facebook o fan page

Facebook pone a disposición de todos sus usuarios el concepto de fanpage con el fin de tener un espacio donde compartir contenido con el resto de usuarios. Un ejemplo de fanpage puede ser la página de un grupo de música, una universidad o un partido político.

Por normal general, cada página tiene una serie de datos que son públicos al resto de usuarios de la red social. Como por ejemplo el número de fans. Esto es el número de usuarios que siguen esta página, y por lo tanto reciben las actualizaciones que en ella se produzcan dentro de su apartado personal, también conocido como muro o feed. Este dato nos interesa y por lo tanto será una de las métricas que usaremos de un modo u otro para llegar a nuestro objetivo. Además se almacenará el avatar de la página para enriquecer los reportes que se generen, así como el país, con el mismo fin.

Publicación

Una publicación es el contenido que un usuario o una fan page comparte con el resto de usuarios, ya sea de forma pública o privada.

[9] LTS o Long Term Support en inglés, indica que la versión tiene soporte a largo plazo por lo que el fabricante asegura un mayor soporte que las versiones estándares.

Cada publicación contiene información de interacción relevante que usaremos en nuestro estudio. De forma visual, a continuación se muestra de una publicación de ejemplo, donde podemos observar la información que ésta nos proporciona:

El resto de usuarios de la red social pueden interaccionar con una publicación de tres formas distintas:

- Escribiendo un comentario, o incluso respondiendo a otro existente.
- Usando los botones de “me gusta”. También conocido como reacciones, donde los usuarios pueden mostrar distintos niveles de afinidad o antipatía con la publicación o incluso con los comentarios
- Compartiendo la publicación, de modo que la publicación aparecerá en el muro del usuario que realiza la acción.

Las reacciones que pueden realizar los usuarios se dividen en 6 opciones como se puede ver a continuación:

Esta opción fue lanzada por Facebook el 24 de Febrero de 2016[10], por lo que para publicaciones anteriores a esta fecha solo podremos tener en cuenta la interacción usando la primera de las reacciones, *Like*.

Algunas reacciones pueden no ser claras o tener doble interpretación como podría ser “Haha” o “Wow”, pues dependerá de cada usuario que lo use para expresar un punto de vista negativo o positivo.

Para el caso de las publicaciones de las fan page, los datos que se consideran relevantes para nuestro estudio son:

- Identificador de la publicación
- Fecha de publicación
- Número de reacciones por tipo (Like, Love, Haha, Wow, Sad, Angry)
- Número de comentarios
- Número de veces que la publicación ha sido compartida

Además, aunque no es estrictamente necesario, se recopilarán los siguientes datos, que pueden ser útiles para una visualización individualizada o incluso posteriores análisis más exhaustivos como reconocimiento del lenguaje natural:

- Mensaje de la publicación (texto, enlace, imagen, vídeo, etc.)
- Descripción

[10] En ciertos países como España e Irlanda esta característica fue lanzada en pruebas en Octubre de 2015, aunque no fue hasta el 24 de Febrero de 2016 cuando estuvo disponible a nivel mundial para el resto de países. https://en.wikipedia.org/wiki/Facebook_like_button

- Tipo de publicación

A su vez, queremos ser capaces de realizar un análisis más profundo por lo que tomaremos todos los comentarios de primer nivel que se hayan generado en cada publicación. Ignorando los comentarios realizados sobre otros comentarios (anidados).

- Identificador del comentario
- Fecha de creación
- Contenido del comentario
- Número de reacciones por tipo (Like, Love, Haha, Wow, Sad, Angry)
- Número de comentarios o réplicas que ha recibido

Ahora que ya hemos acabado de describir los conceptos necesarios para la aplicación, pasaremos a la parte de requisitos que debe cumplir la misma.

3.2. Requisitos

La aplicación permitirá automatizar los procesos de recopilación, almacenamiento, análisis y visualización de los datos desde un navegador web.

Django nos provee de forma automática con un panel de administración desde el cual se pueden realizar operaciones con los modelos de datos que hayamos generado. Por lo tanto desde el principio se contará con una interfaz web básica desde la cual podremos insertar las fan pages de los partidos políticos que nos interese estudiar.

3.3. Estructura de datos

En nuestra aplicación no tenemos ninguna necesidad especial, por lo que no hay ninguna limitación a la hora de usar un sistema de gestión de base de datos específico. Se ha decidido usar PostgreSQL 9.5.11, para lo que Django tiene soporte nativo para este sistema de gestión de bases de datos relacional orientado a objetos.

Nuestro esquema de almacenamiento será básicamente dos modelos o tablas: Page y Post.

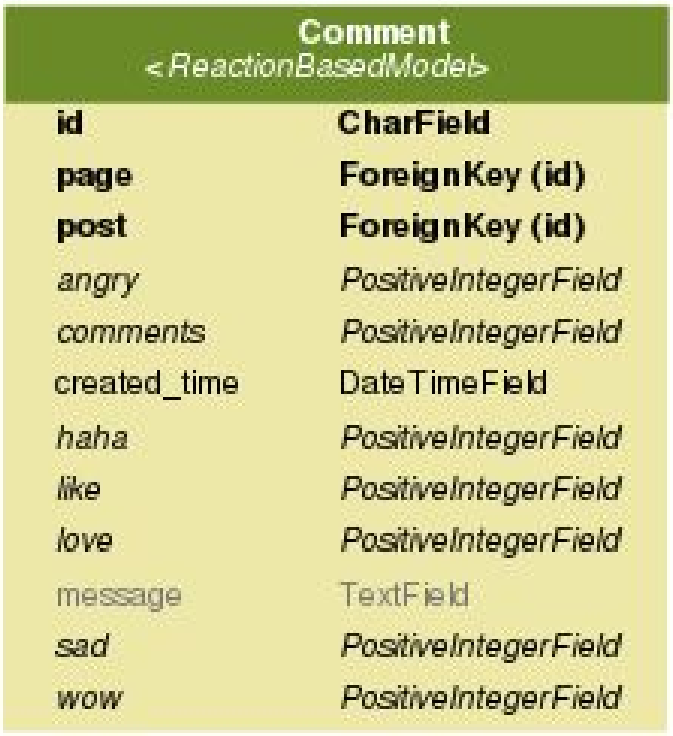
A continuación se muestra el esquema de la tabla Page, donde se almacenarán los datos relacionados con las páginas involucradas en nuestro estudio:

Page	
id	**BigAutoField**
country	CharField
fan_count	PositiveIntegerField
name	CharField
picture	URLField

Por otro lado, Post, donde se almacenará lo relacionado con publicaciones. Cuenta con una relación con la tabla Page mediante una clave externa en Post.page.

Post *<ReactionBasedModel>*	
id	**CharField**
page	**ForeignKey (id)**
angry	*PositiveIntegerField*
comments	*PositiveIntegerField*
created_time	DateTimeField
description	TextField
haha	*PositiveIntegerField*
like	*PositiveIntegerField*
love	*PositiveIntegerField*
message	TextField
sad	*PositiveIntegerField*
shares	PositiveIntegerField
type	CharField
wow	*PositiveIntegerField*

Finalmente la tabla Comment, que contiene una relación dos claves externas. La primera Comment.page hacia la tabla Page y la segunda Comment.post hacia la tabla Post. De esta forma podremos agilizar y reducir las consultas a la base de datos para obtener el conjunto de comentarios relacionados con ciertas publicaciones o páginas.



Los gráficos anteriores han sido generados de forma separada con Graph Models[11], incluída en la libreria django_extensions[12], una herramienta muy útil para generar esquemas gráficos de las estructuras de datos de un proyecto en Django.

Recopilación de datos

Para la recopilación de datos la aplicación contará con una serie de comandos que se podrán ejecutar a demanda para capturar datos en lote sobre las publicaciones, y que sea capaz de correr en segundo plano pues será un proceso que llevará algunos minutos, en función del rango de fechas que se especifique. Hablaremos de este procedimiento a continuación.

[11] Graph Models: http://django-extensions.readthedocs.io/en/stable/graph_models.html
[12] Django extensions: http://django-extensions.readthedocs.io/en/stable/index.html

Facebook pone a disposición su API, que usaremos de forma intensa para la extracción de todos los datos de los que hablamos con anterioridad, páginas y publicaciones. Hablaremos más profundamente sobre la API de Facebook en el capítulo siguiente, de forma que podemos centrarnos en describir cómo funcionará el extractor sin entrar en detalle a hablar sobre dicha API.

Nuestro extractor tomará 4 parámetros de entrada, estos son:

- Token de acceso, necesario para la autenticación en Facebook
- Identificador de la página de la que deseamos extraer la información
- Rangos de fecha inicial y final donde se debe buscar las publicaciones en formato YYYY-MM-DD (año, mes y día)

Ejemplo de ejecución del comando:

```
python manage.py extract --page=215062516098 --start_date=2017-12-01
        --end_date=2018-01-15 --get_posts --get_comments --access_token=1234
```

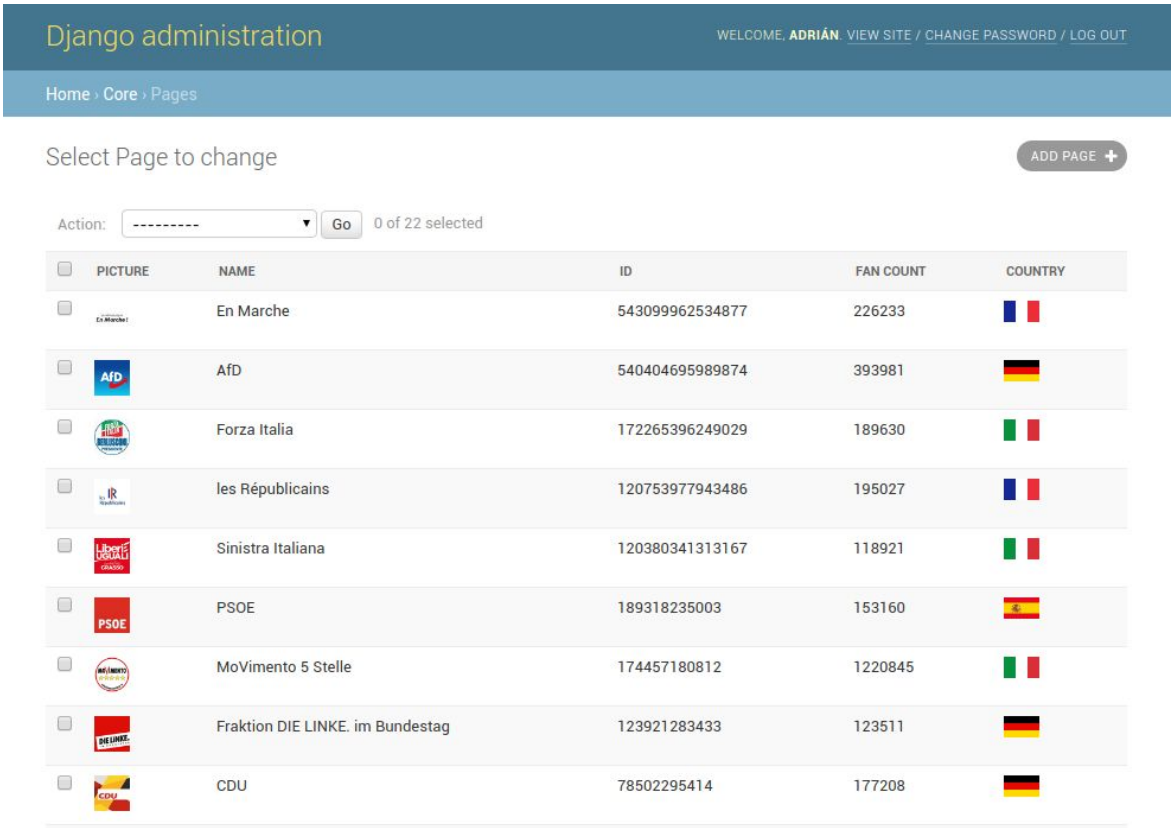
Como se puede ver en el comando de ejemplo tenemos 6 parámetros de entrada, aunque solo 4 de ellos son obligatorios. Estos parámetros obligatorios son *--page*, *--start_date*, *--end_date* y *--access_token*. Siendo los parámetros --get_posts y --get_comments opcionales.

Para este ejemplo hemos usado el identificador de la página que la Universidad de La Laguna tiene en Facebook (page) para un rango de fechas entre el 1 de Diciembre de 2018 y el 15 de Enero de 2018.

El token de acceso ha sido eliminado y sustituido por una cadena aleatoria, pues la longitud de este dato está en torno a los 200 caracteres y no es relevante en este contexto.

El primer procedimiento que lleva a cabo el comando es la extracción de datos de la página, por tanto tras la ejecución de este paso tendremos a disposición los datos procedentes de esa página.

Después de introducir varios partidos políticos podremos ver en el panel de administración web de Django los datos extraídos como se muestra en la imagen a continuación. Los datos están ordenados en orden alfabético:



Indicar además que el parámetro `--page` aceptará varios identificadores separados por coma, de este modo podemos extraer datos de varias páginas al mismo tiempo.

Es muy importante hablar sobre el proceso de recopilación de las páginas de Facebook que van a ser sometidas al estudio y cuáles y cómo han sido seleccionadas para su extracción, hablaremos de esto posteriormente en el capítulo 6.

Continuando con los requisitos de nuestra aplicación, el extractor procederá si el parámetro `--get_posts` ha sido suministrado, en cuyo caso se realizará la extracción de cada uno de las publicaciones que se han realizado durante ese periodo de fechas especificado entre los parámetros `--start_date` y `--end_date`. Normalmente las páginas de partidos políticos no permiten que usuarios externos escriban publicaciones directamente en el feed de la página, pues podría suponer un problema de imagen. En cualquier caso, durante este proceso se ignorarán, si las hubiera, las publicaciones que hayan sido realizadas por los usuarios.

Para cada publicación extraída el software hace una extracción más profunda, revisando en este caso los comentarios realizados en la publicación concreta en la que nos encontramos, siempre y cuando el parámetro `--get_comments` haya sido suministrado.

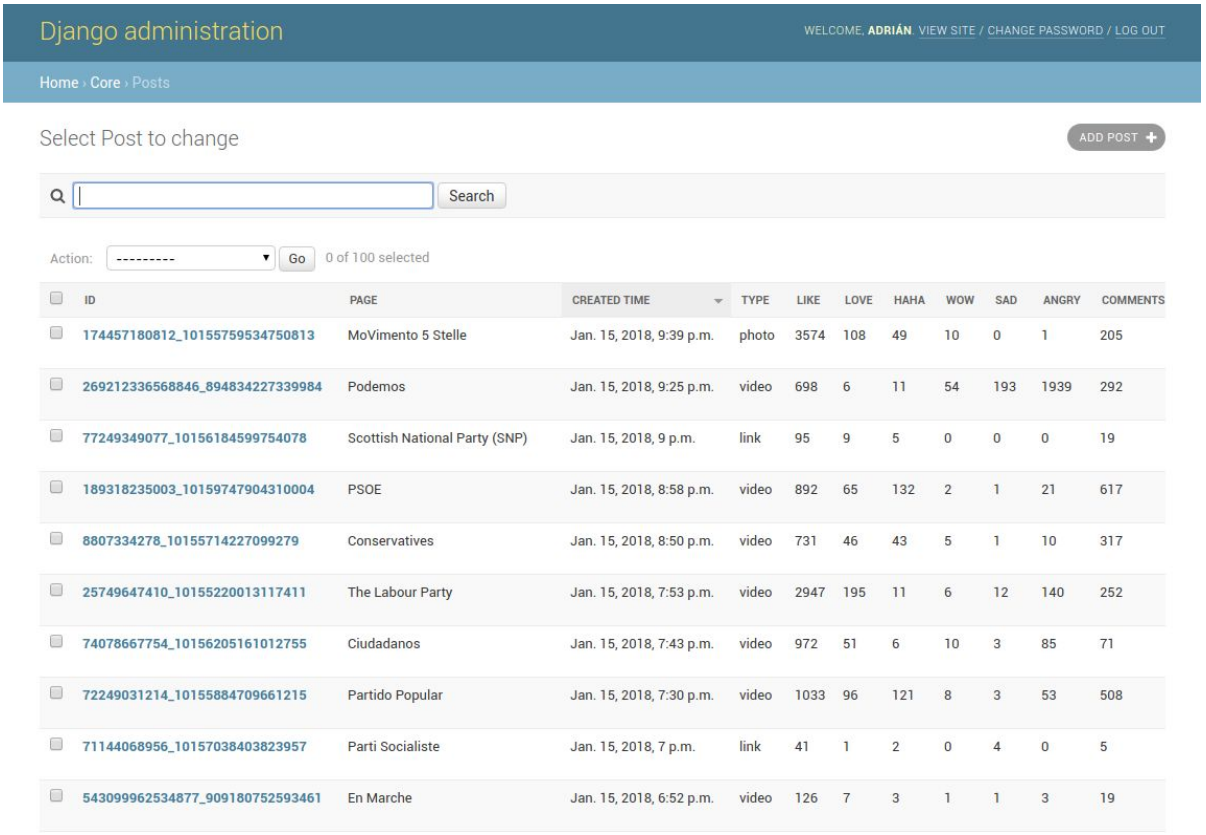
La relación en Facebook entre páginas, publicaciones y comentarios es la siguiente:

$$\text{Página } (1 \rightarrow N) \text{ Publicaciones}$$
$$\text{Publicación } (1 \rightarrow M) \text{ Comentarios}$$

Esto implica que la tabla Comment crecerán de forma exponencial tan pronto las publicaciones y las páginas vayan creciendo en el resto de tablas:

El proceso de extracción es muy costoso, pues se requiere hacer un gran número de llamadas a la API no solo porque se trate de un gran volumen de datos, que lo es, si no también debido a que la API de Facebook usa paginación para dar un conjunto de resultados. Esto es básicamente trocear el resultado y suministrar la respuesta en varias piezas. Por defecto la API de Facebook suministra un lote de 25 items por página, lo que implica que especialmente durante la campaña electoral, se necesiten realizar demasiadas llamadas a la API pudiendo incurrir en el límite máximo de peticiones por tiempo o por IP. Se hablará de estas limitaciones en el capítulo X.

Desde el panel de administración de Django también se tendrá acceso a las publicaciones y comentarios. En la siguiente imagen se muestra las publicaciones de diferentes partidos políticos a modo de ejemplo, ordenados cronológicamente, últimas publicaciones primero:



ID	PAGE	CREATED TIME	TYPE	LIKE	LOVE	HAHA	WOW	SAD	ANGRY	COMMENTS
174457180812_10155759534750813	MoVimento 5 Stelle	Jan. 15, 2018, 9:39 p.m.	photo	3574	108	49	10	0	1	205
269212336568846_894834227339984	Podemos	Jan. 15, 2018, 9:25 p.m.	video	698	6	11	54	193	1939	292
77249349077_10156184599754078	Scottish National Party (SNP)	Jan. 15, 2018, 9 p.m.	link	95	9	5	0	0	0	19
189318235003_10159747904310004	PSOE	Jan. 15, 2018, 8:58 p.m.	video	892	65	132	2	1	21	617
8807334278_10155714227099279	Conservatives	Jan. 15, 2018, 8:50 p.m.	video	731	46	43	5	1	10	317
25749647410_10155220013117411	The Labour Party	Jan. 15, 2018, 7:53 p.m.	video	2947	195	11	6	12	140	252
74078667754_10156205161012755	Ciudadanos	Jan. 15, 2018, 7:43 p.m.	video	972	51	6	10	3	85	71
72249031214_10155884709661215	Partido Popular	Jan. 15, 2018, 7:30 p.m.	video	1033	96	121	8	3	53	508
71144068956_10157038403823957	Parti Socialiste	Jan. 15, 2018, 7 p.m.	link	41	1	2	0	4	0	5
543099962534877_909180752593461	En Marche	Jan. 15, 2018, 6:52 p.m.	video	126	7	3	1	1	3	19

3.4. Cálculo de resultados

En una segunda fase, la aplicación deberá ser capaz de generar una predicción del número de votos, para ellos se tendrán en cuenta los datos cuantitativos que se han extraído en este capítulo, así como los datos de análisis de sentimiento de textos de cada publicación, del que hablaremos en el capítulo 5, haciendo uso de dos herramientas de lenguaje natural.

4. APIS

API o *Application programming interface*, por sus siglas en inglés, es un conjunto de subrutinas, protocolos y herramientas para construir aplicaciones de software. En términos generales, es un conjunto de métodos de comunicación bien definidos entre varios componentes de software. Una buena API hace más fácil el desarrollo poniendo a disposición del programador todos los bloques necesarios para la construcción de la aplicación.

Una API puede existir en una aplicación web, en un sistema operativo, un sistema de base de datos, un componente de hardware o una librería de software.

Las especificaciones de una API pueden tomar distintas formas, pero normalmente incluyen especificaciones para sus rutinas, estructuras de datos, clases de objetos, variables y subrutinas remotas ejecutadas en otro computador o red distribuida.

Las APIs pueden ser categorizadas de la siguiente manera:

- API de librerías y frameworks
- API del sistema operativo
- API remota
- API web

La API de Facebook se encuentra en la categoría de APIs web, que se describe a continuación.

4.1. API web

Las APIs web son interfaces a través de las cuales se realiza una interacción entre una compañía y una aplicación que hace uso de sus recursos. Un enfoque API es un enfoque arquitectónico que gira en torno al suministro de interfaces programables de un conjunto de servicios para diferentes aplicaciones que atienden a diferentes tipos de consumidores.

Cuando se usa en el contexto del desarrollo web, una API generalmente se define como un conjunto de solicitudes HTTP (Hypertext Transfer Protocol)[13], junto con una definición de la estructura de la respuesta normalmente en formato Extensible Markup Language (XML)[14] o JSON (JavaScript Object Notation)[15].

Como ejemplo podemos pensar en la API de una empresa de mensajería que puede ser conectada con una tienda online para facilitar los servicios de envío e incluir el costo de los envíos sin necesidad de incluir los datos del proveedor en la misma base de datos de la tienda online y actualizar estos precios cada vez que el proveedor cambie alguna tarifa de envío.

Las API web permiten la combinación de múltiples APIs en nuevas aplicaciones conocidas como mashups[16]. En el espacio de las redes sociales, las APIs web han permitido que las comunidades web faciliten el intercambio de contenido y datos entre comunidades y aplicaciones. De esta forma, el contenido que se crea en un lugar se puede publicar y actualizar dinámicamente en varias ubicaciones en la web.

En la web, las API's son publicadas por sitios para brindar la posibilidad de realizar alguna acción o acceder a alguna característica o contenido que el sitio provee. Algunos ejemplos de servicios de APIs abiertas son:

Servicio	**Documentación API**
Flickr	https://www.flickr.com/services/api
Twitter	https://developer.twitter.com/en/docs/api-reference-index
Facebook	https://developers.facebook.com/docs/graph-api
Google Maps	https://developers.google.com/maps/web
Amazon	https://developer.amazon.com/services-and-apis

[13] Hypertext Transfer Protocol
[14] Extensible Markup Language (XML). https://es.wikipedia.org/wiki/Extensible_Markup_Language
[15] Javascript Object Notation (JSON). https://es.wikipedia.org/wiki/JSON
[16] Niccolai, James (2008-04-23), "So What Is an Enterprise Mashup, Anyway?", PC World

4.2. API de Facebook

Facebook, al igual que otras muchas empresas, ponen a disposición de los desarrolladores una API abierta con la que poder realizar cierta extracción.

Se trata del pilar central, gracias al cual es posible la extracción y automatización de los datos. Sin ella no se dispondría de los datos para llevar el estudio de este proyecto.

La Graph API es el modo principal para extraer e ingresar datos en la plataforma de Facebook. Se trata de una API basada en HTTP de nivel inferior que puedes utilizar de manera programática para consultar datos, publicar nuevas historias, administrar anuncios, subir fotos y llevar a cabo varias tareas más propias de una aplicación.

La Graph API se llama así por la idea de una "gráfica social", una representación de la información en Facebook que consta de:

- **Nodos:** básicamente elementos como un usuario, una foto, una página, un comentario
- **Perímetros:** las conexiones entre esos elementos, como fotos de páginas o comentarios de fotos
- **Campos:** información sobre esos elementos, como el cumpleaños de una persona o el nombre de una página

La Graph API está basada en HTTP, por lo que funciona con cualquier lenguaje que tenga una librería HTTP, como cURL y urllib.

4.3. Autenticación de la API de Facebook

Las solicitudes que se realizan a la Graph API necesitan el uso de tokens de acceso que se pueden generar implementando el inicio de sesión con Facebook.

Para ello es necesario haber generado una aplicación de inicio de sesión en la página de Facebook Developers (https://developers.facebook.com) y disponer de un

identificador de la aplicación y una clave secreta de la aplicación. De este modo, y mediante el uso del kit de desarrollo (Software Development Kit o más comunmente SDK por sus siglas en inglés) de Python para Facebook Graph API. Dicho SDK se encuentra disponible en https://github.com/mobolic/facebook-sdk.

Con motivo de agilizar el desarrollo se omitirá el uso del SDK y se usará la herramienta que Facebook pone a disposición, Graph API Explorer, disponible en la dirección web: https://developers.facebook.com/tools/explorer/. Esta herramienta permite realizar llamadas a la API de una forma visual y rápido acceso a los campos de cada endpoint, lo que permite, en muchos casos, usar endpoints sin necesidad de acudir a la documentación de la propia API. Desde esta herramienta seremos capaces de generar un token de usuario que será usado posteriormente desde los diferentes scripts de recogida de datos.

Los tokens de acceso generados tienen una caducidad que puede variar en función del tipo de token generado.

4.4. Endpoints usados

A continuación se procede a describir los tres nodos o endpoints que son relevantes para este proyecto y han sido usados para la extracción de los datos como se describen a continuación.

Page

Representa una página de Facebook. El nodo **/{page-id}** devuelve una sola página, donde **{page-id}** es el identificador de la página que quiere ser consultada.

Opcionalmente se puede indicar los campos que deseamos obtener. En nuestro caso hemos usado los siguientes:

- id
- name
- location
- fan_count
- picture{{url}}

Más información: https://developers.facebook.com/docs/graph-api/reference/page

Feed

El feed de publicaciones (incluyendo actualizaciones de estado) y enlaces publicados por una página o por terceros en dicha página. Se accede mediante el nodo **/{page-id}/posts**, dónde {page-id} representa el identificador de la página cuyas publicaciones se desean extraer.

Más información:
https://developers.facebook.com/docs/graph-api/reference/page/feed

Al igual que en el anterior endpoint, se puede indicar los campos que deseamos obtener. En nuestro caso hemos usado los siguientes:

- id
- created_time
- message
- decription
- from
- type
- reactions.type(LIKE).limit(0).summary(total_count).as(like)
- reactions.type(LOVE).limit(0).summary(total_count).as(love)
- reactions.type(HAHA).limit(0).summary(total_count).as(haha)
- reactions.type(WOW).limit(0).summary(total_count).as(wow)
- reactions.type(SAD).limit(0).summary(total_count).as(sad)
- reactions.type(ANGRY).limit(0).summary(total_count).as(angry)
- shares.limit(0).summary(total_count)
- comments.limit(0).summary(total_count)

Más información: https://developers.facebook.com/docs/graph-api/reference/post

Comments

Representa los comentarios para un objeto como post, album, photo, status, link, etc.
Su uso se realiza a través del modo **/{object-id}/comments**, dónde {object-id} puede es el identificador para post, album, status, link o incluso comment pues un comentario puede tener comentarios asociados. Para este endpoint se usan los mismos campos que en el caso de Post.

Más información:
https://developers.facebook.com/docs/graph-api/reference/object/comments

4.5. Limitaciones de la API de Facebook

El mayor inconveniente con el que contamos a la hora de realizar la extracción es la limitación de llamada a la API. Dado que vamos a extraer multitud de datos, tendremos que realizar las llamadas a la API con algún que otro tiempo de espera entre dichas llamadas.
Más información sobre límites de la API se puede consultar el siguiente enlace: https://developers.facebook.com/docs/graph-api/advanced/rate-limiting donde se muestran cifras de ejemplo del número máximo de llamadas a la API en función del número de usuarios de la aplicación.

Cabe indicar, además, que desde el escándalo de Cambridge Analytica[17] [18], Facebook a realizado cambios en la API respecto a la privacidad de los usuarios. Aunque este hecho no es relevante para el estudio de este trabajo, antes de dichos cambios, era posible extraer el nombre y apellido del usuario que realizaba una publicación o comentario, desde hace algunos meses esto ya no es posible y los datos se muestran anonimizados.

Al término de este trabajo, para las páginas de Facebook de partidos españoles, se han extraído un total 12.892 publicaciones y 268.781 comentarios.

Con el objetivo de tener una visión general del número de llamas a la API que son necesarias para nuestro estudio se muestra la siguiente tabla de ejemplo para diferentes número de páginas, publicaciones y comentarios.

Ejemplo	N° páginas	N° publicaciones	N° comentarios por publicación	Total llamadas a la API
A	1	1	1	3
B	1	1	10	3
C	1	1	100	6
D	10	10	100	510

[17] https://www.theguardian.com/news/2018/mar/26/the-cambridge-analytica-files-the-story-so-far
[18] https://es.wikipedia.org/wiki/Cambridge_Analytica

Para cada ejemplo se realizan las siguientes llamadas:

1. **/{page-id}** donde se extrae la información de la página
2. **/{page-id}/posts** donde se extrae la información de las publicaciones
3. **/{post-id}/comments** para extraer la información de los comentarios. En esta llamada solo se pueden extraer 25 comentarios máximos, lo que implica que para el ejemplo C y D se deban realizar 4 llamadas para cada publicación, con el fin de obtener los 100 comentarios.

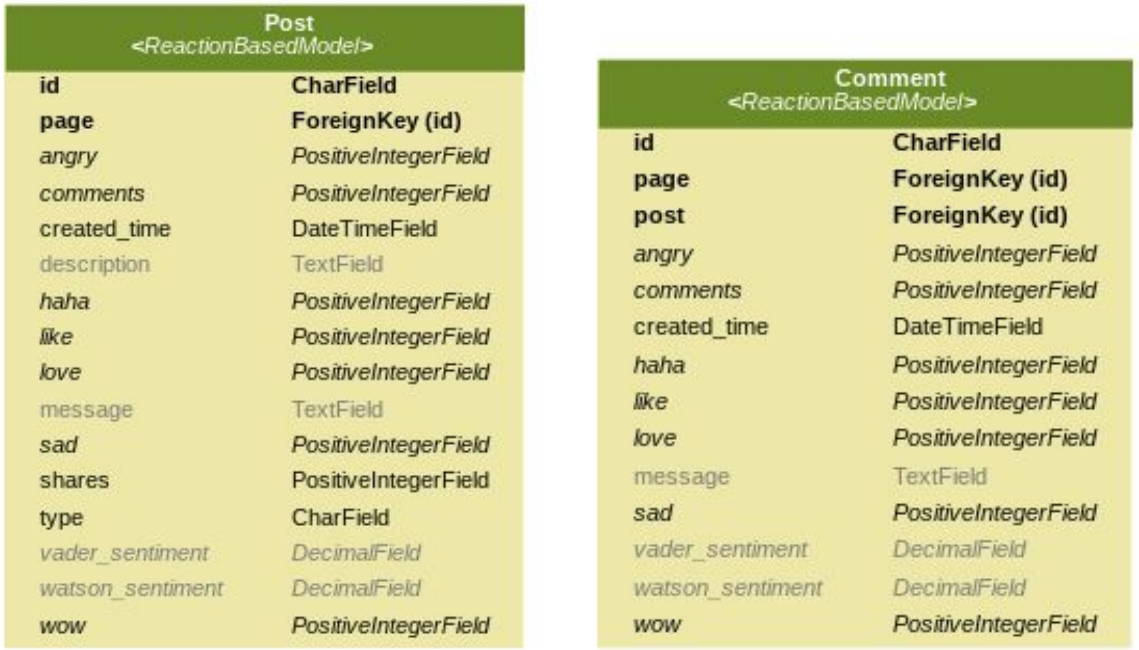
5. Lenguaje natural

Con el fin de tener más información y de mejor calidad se pretende usar herramientas de análisis de lenguaje natural, concretamente análisis de sentimiento.

En el mercado existen diferentes utilidades para realizar análisis de textos con el fin de extraer información relacionada con el sentimiento. Durante la realización de este trabajo se han utilizado concretamente dos enfoques, uno con Natural Language Understanding (Watson), de IBM, y otro enfoque haciendo uso de la librería open source Natural Language Toolkit sobre Python.

5.1. Cambios en base de datos

Con el fin de almacenar el valor del sentimiento para cada publicación y cada comentario, se han modificado las tablas Post y Comment, añadiendo dos nuevos campos: *watson_sentiment* y *vader_sentiment*, como se muestra en el siguiente gráfico.



5.2. Natural Language Understanding (NLU, Watson)

Natural Language Understanding es una colección de APIs que ofrece análisis de texto a través del procesamiento de lenguaje natural. Este conjunto de APIs puede analizar texto para ayudar en la comprensión de sus conceptos, entidades, palabras

clave, sentimiento, etc.. Además, puede crear un modelo personalizado para algunas API para obtener resultados específicos que se adaptan a un dominio concreto.

Watson soporta análisis para distintos idiomas: árabe, inglés, francés, alemán, italiano, japonés, coreano, portugués, ruso y español.

Se trata de un servicio de pago por suscripción, aunque cuentan con cuentas demo para estudiantes, lo cual hizo posible contar con esta herramienta sin que supusiera coste alguno.

Este plan "Lite" incluye 30.000 NLU (Natural Language Units) por mes, así como un modelo personalizado[19].

Cada NLU se basa en la cantidad de unidades de datos enriquecidas y la cantidad de características de enriquecimiento aplicadas. Una unidad de datos tiene 10,000 caracteres o menos. Por ejemplo: la extracción de Entidades y Sentimientos de 15,000 caracteres de texto es (2 Unidades de Datos * 2 Características de Enriquecimiento) = 4 Artículos de NLU.

Uso de la API

Existe una librería para Python, watson-developer-cloud[20], que nos permite hacer uso de los servicios de Watson cloud de manera extremadamente sencilla. Esta librería está disponible en el siguiente enlace, en el repositorio de Github de IBM:

https://github.com/watson-developer-cloud/python-sdk

Así como la documentación para cada uno de los diferentes servicios de IBM en relación con Natural Language Understanding:

https://www.ibm.com/watson/services/natural-language-understanding/

Para cada petición de análisis de sentimiento, la API de Watson nos devuelve un valor entre -1 y 1, en función de si el sentimiento es negativo, neutral o positivo.

A continuación se muestra un ejemplo de como realizar las llamas a la API de Watson usando la librería watson_developer_cloud.

[19] Un modelo personalizado se refiere a un modelo de anotación desarrollado con Watson Knowledge Studio.

[20] https://pypi.org/project/watson-developer-cloud/

```
      from watson_developer_cloud import NaturalLanguageUnderstandingV1
      from watson_developer_cloud.natural_language_understanding_v1 import (
          Features, SentimentOptions
      )
      natural_language_understanding = NaturalLanguageUnderstandingV1(
          username=USERNAME, password=PASSWORD, version=VERSION)
      natural_language_understanding.analyze(
          text=MESSAGE,
          language=LANG,
          features=Features(sentiment=SentimentOptions()))
```

Y a continuación se muestra la salida (JSON):

```
{
  "usage": {
       "text_units": 1,
       "text_characters": 38,
       "features": 1
  },
  "sentiment": {
       "document": {
       "score": 0.981942,
       "label": "positive"
       }
  },
  "language": "es"
}
```

Retos con el uso de Watson

El principal reto con el que se ha tenido que lidiar ha sido con la cantidad de datos y por consiguiente el alcance de los límites de la cuenta demo gratuita. Por ello fue necesario realizar una primera extracción y esperar que los límites de dicha cuenta fueran reseteados al finalizar el mes en curso.

Por otro lado, se comenzó realizando la extracción de publicaciones y comentarios, pero posteriormente se decidió analizar únicamente el sentimiento de las publicaciones, pues de otra forma se volvería a alcanzar los límites de la cuenta al tratarse de un servicio de pago.

5.3. Natural Language Toolkit (NLT)

Se trata de una librería open source que funciona sobre Python. A diferencia de Watson, este sistema no se ejecuta en sistemas de terceros y tampoco funciona con un sistema de suscripción. Basta con importar las librerías necesarias para ejecutarlo en nuestros propios sistemas.

Para el estudio se ha utilizado, concretamente, Vader Sentiment Analyzer[21], que al igual que Watson, nos proporciona un valor entre -1 y 1 para indicar si el sentimiento para un texto es negativo, neutral o positivo.

A continuación se muestra, a modo de ejemplo, una porción de código usado en la aplicación web que se ha realizado:

```
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

analyser = SentimentIntensityAnalyzer()
result = analyser.polarity_scores("I enjoy traveling around the world")
```

Y como salida, en la variable result, se obtiene el siguiente resultado:

```
{'neg': 0.0, 'neu': 0.556, 'pos': 0.444, 'compound': 0.4939}
```

Limitaciones de NLT

La mayor limitación con la que nos encontramos haciendo uso de esta librería es que solo soporta inglés como idioma de entrada, por lo que no podremos tener en cuenta este valor para países de habla no inglesa.

5.4. Automatización del proceso

Para realizar el análisis de sentimiento de los datos obtenidos de Facebook se ha realizado un script para automatizar el proceso, permitiendo así el análisis de publicaciones y/o comentarios para una fecha concreta y haciendo uso de uno u otro método (Watson o NLT-Vader).

[21] Hutto, C.J. & Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014.

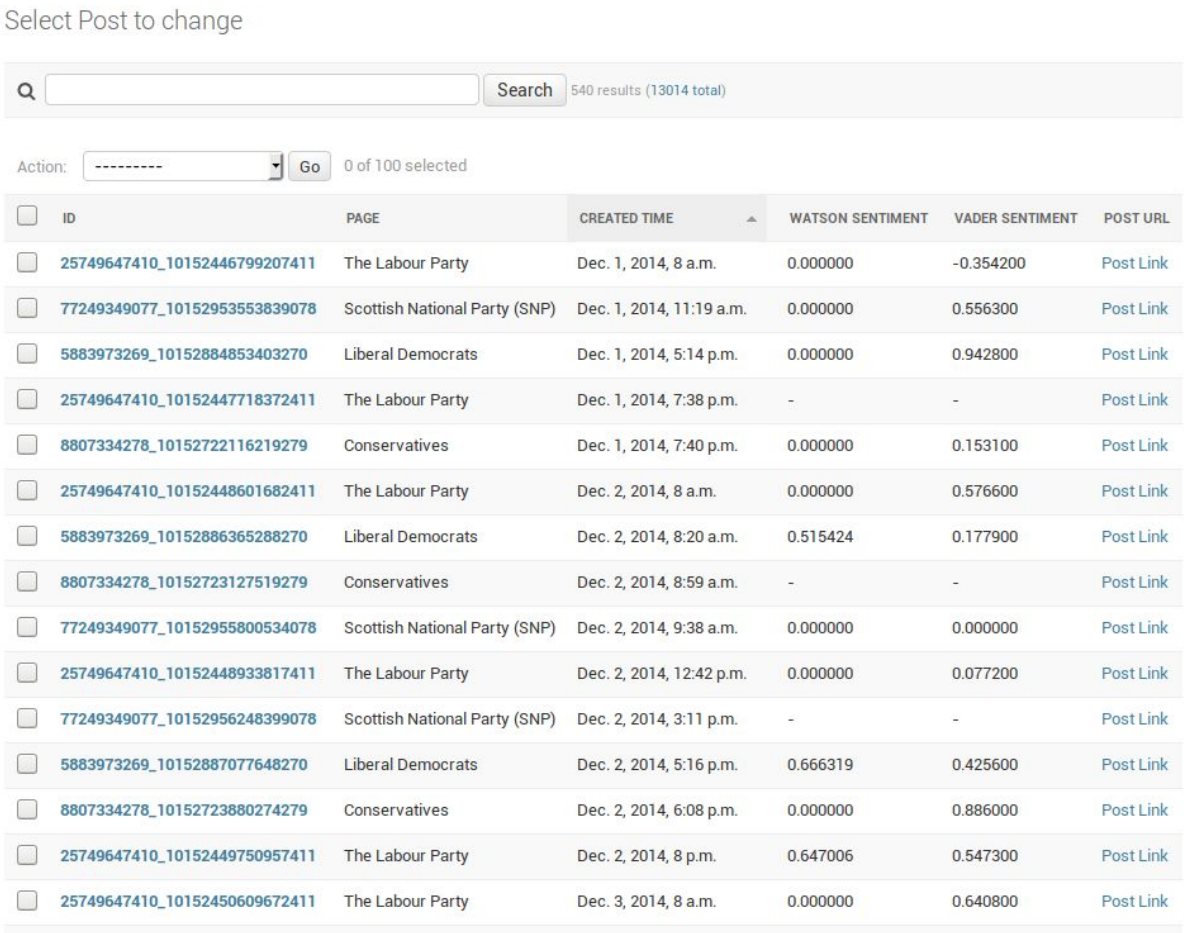
```
$ python manage.py analyze --use-watson --country=Spain -s 2014-12-01 -e
2018-06-23
       --posts --comments
```

```
$ python manage.py analyze --use-vader --country=Spain -s 2014-12-01 -e
2018-06-23
       --posts --comments
```

En los ejemplos que se observan se realiza el análisis para publicaciones y comentarios de partidos políticos españoles en el rango de fechas comprendidos entre el 1 de Diciembre de 2014 y 23 de Junio de 2018

Para cada publicación y cada comentario el software actualiza cada registro de la base de datos para incorporar el valor de sentimiento en las columnas *watson_sentiment* y *vader_sentiment* según el se use el parámetro *--use-watson* o *--use-vader*.

De esta forma, tal como se puede ver en el panel de administración de Django, el valor del sentimiento tanto usando Watson, como Vader, recibe valores entre -1 y 1, tal como se mencionó posteriormente.



ID	PAGE	CREATED TIME	WATSON SENTIMENT	VADER SENTIMENT	POST URL
25749647410_10152446799207411	The Labour Party	Dec. 1, 2014, 8 a.m.	0.000000	-0.354200	Post Link
77249349077_10152953553839078	Scottish National Party (SNP)	Dec. 1, 2014, 11:19 a.m.	0.000000	0.556300	Post Link
5883973269_10152884853403270	Liberal Democrats	Dec. 1, 2014, 5:14 p.m.	0.000000	0.942800	Post Link
25749647410_10152447718372411	The Labour Party	Dec. 1, 2014, 7:38 p.m.	-	-	Post Link
8807334278_10152722116219279	Conservatives	Dec. 1, 2014, 7:40 p.m.	0.000000	0.153100	Post Link
25749647410_10152448601682411	The Labour Party	Dec. 2, 2014, 8 a.m.	0.000000	0.576600	Post Link
5883973269_10152886365288270	Liberal Democrats	Dec. 2, 2014, 8:20 a.m.	0.515424	0.177900	Post Link
8807334278_10152723127519279	Conservatives	Dec. 2, 2014, 8:59 a.m.	-	-	Post Link
77249349077_10152955800534078	Scottish National Party (SNP)	Dec. 2, 2014, 9:38 a.m.	0.000000	0.000000	Post Link
25749647410_10152448933817411	The Labour Party	Dec. 2, 2014, 12:42 p.m.	0.000000	0.077200	Post Link
77249349077_10152956248399078	Scottish National Party (SNP)	Dec. 2, 2014, 3:11 p.m.	-	-	Post Link
5883973269_10152887077648270	Liberal Democrats	Dec. 2, 2014, 5:16 p.m.	0.666319	0.425600	Post Link
8807334278_10152723880274279	Conservatives	Dec. 2, 2014, 6:08 p.m.	0.000000	0.886000	Post Link
25749647410_10152449750957411	The Labour Party	Dec. 2, 2014, 8 p.m.	0.647006	0.547300	Post Link
25749647410_10152450609672411	The Labour Party	Dec. 3, 2014, 8 a.m.	0.000000	0.640800	Post Link

Las filas con valores vacíos se deben a que dichas publicaciones no cuentan con un mensaje, pues la publicación puede ser de tipo foto o enlace por ejemplo. Nótese, además, que los resultados que se muestran son para partidos políticos de Reino Unido, lo cual hace posible realizar el análisis de sentimiento para las publicaciones, pues el idioma de las mismas es inglés.

5.5 Watson vs NLT

Como puede verse en la anterior captura de pantalla ambos valores difieren completamente.

NLT es una librería open source y gratuita, lo que a priori hace que este sea el método que se use, descartando Watson, que conlleva un gasto adicional para el caso de querer ampliar el análisis a todos las publicaciones y comentarios y querer profundizar en posteriores estudios.

Sin embargo, lamentablemente debido a que NLT no tiene soporte para idioma español, la decisión es usar Watson. De este modo el estudio se centra únicamente

en el análisis de sentimientos para publicaciones, y abarcar el 100% de la muestra de datos sin incurrir en limitaciones de la API de IBM.

6. Toma de datos

En este punto, nuestra aplicación web contiene todas las funcionalidades necesarias para la extracción de datos, estos son:

- Extracción de datos de páginas de Facebook
- Extraer publicaciones
- Extraer comentarios
- Realizar análisis de sentimientos para publicaciones/comentarios

6.1. Selección de partidos políticos

La aplicación web permite almacenar, manipular así como filtrar para un determinado país. Por ello como primer paso se realiza la selección de países y partidos políticos más relevantes a nivel nacional.

Es importante aclarar la importancia de seleccionar correctamente la página de Facebook que va a ser objeto de estudio para cada partido. En los meses previos a las elecciones, cada partido decide crear una página exclusiva para publicitar el candidato, haciendo que estas páginas tenga una validez únicamente durante la campaña electoral concreta y además algunas otras son eliminadas o abandonadas. Es esta la razón por la que se decidió utilizar en cada caso la página oficial de cada partido y no la del candidato.

A continuación se muestra un listado con los partidos seleccionados, organizados por país y ordenados alfabéticamente. Entre paréntesis se indica el identificador usado en Facebook:

Alemania 	AfD (alternativefuerde)
	BÜNDNIS 90/DIE GRÜNEN (B90DieGruenen)
	CDU (CDU)
	FDP (FDP)
	Fraktion DIE LINKE. im Bundestag (linksfraktion)
	SPD (SPD)

España 	Ciudadanos (Cs.Ciudadanos)
	En Comú Podem (encomupodem)
	PSOE (psoe)
	Partido Popular (pp)
	Podemos (ahorapodemos)

Francia 	En Marche (EnMarche)
	Parti Socialiste (partisocialiste)
	Parti de Gauche (partidegauche.national)
	Rassemblement National (RassemblementNational)
	les Républicains (les.Republicains.FR)

Italia 	Forza Italia (ForzaItaliaUfficiale)
	MoVimento 5 Stelle (movimentocinquestelle)
	Partito Democratico (partitodemocratico)
	Sinistra Italiana (sinistraitalianaSI)

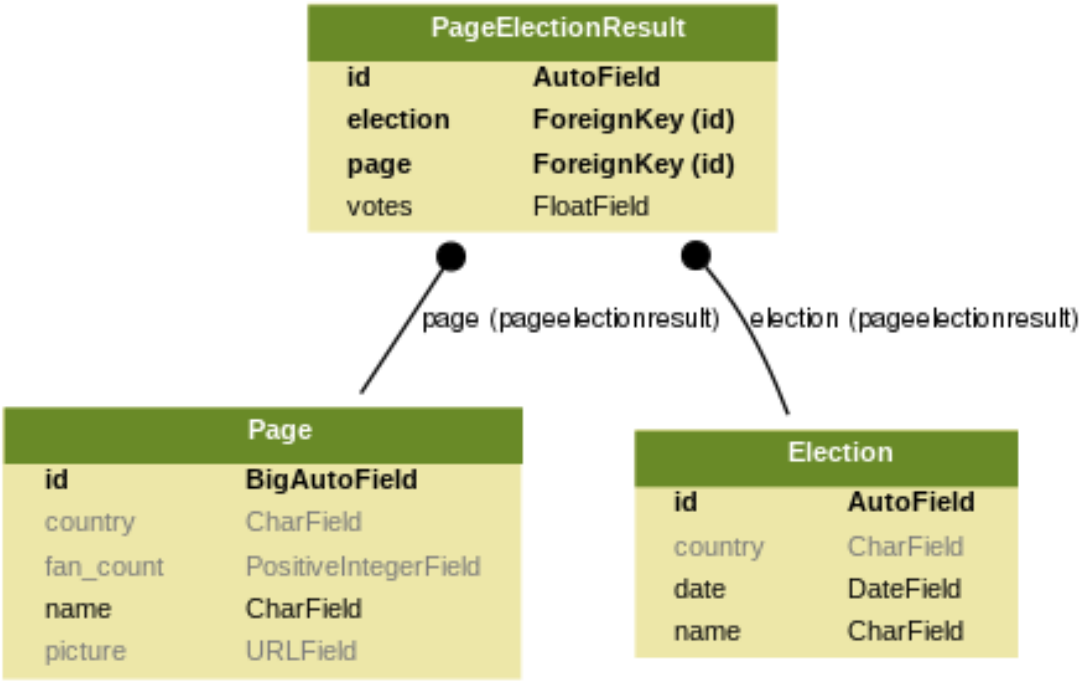
Reino Unido 	Conservatives (conservatives)
	Liberal Democrats (libdems)
	Scottish National Party (theSNP)
	The Labour Party (labourparty)

6.2. Periodos de extracción

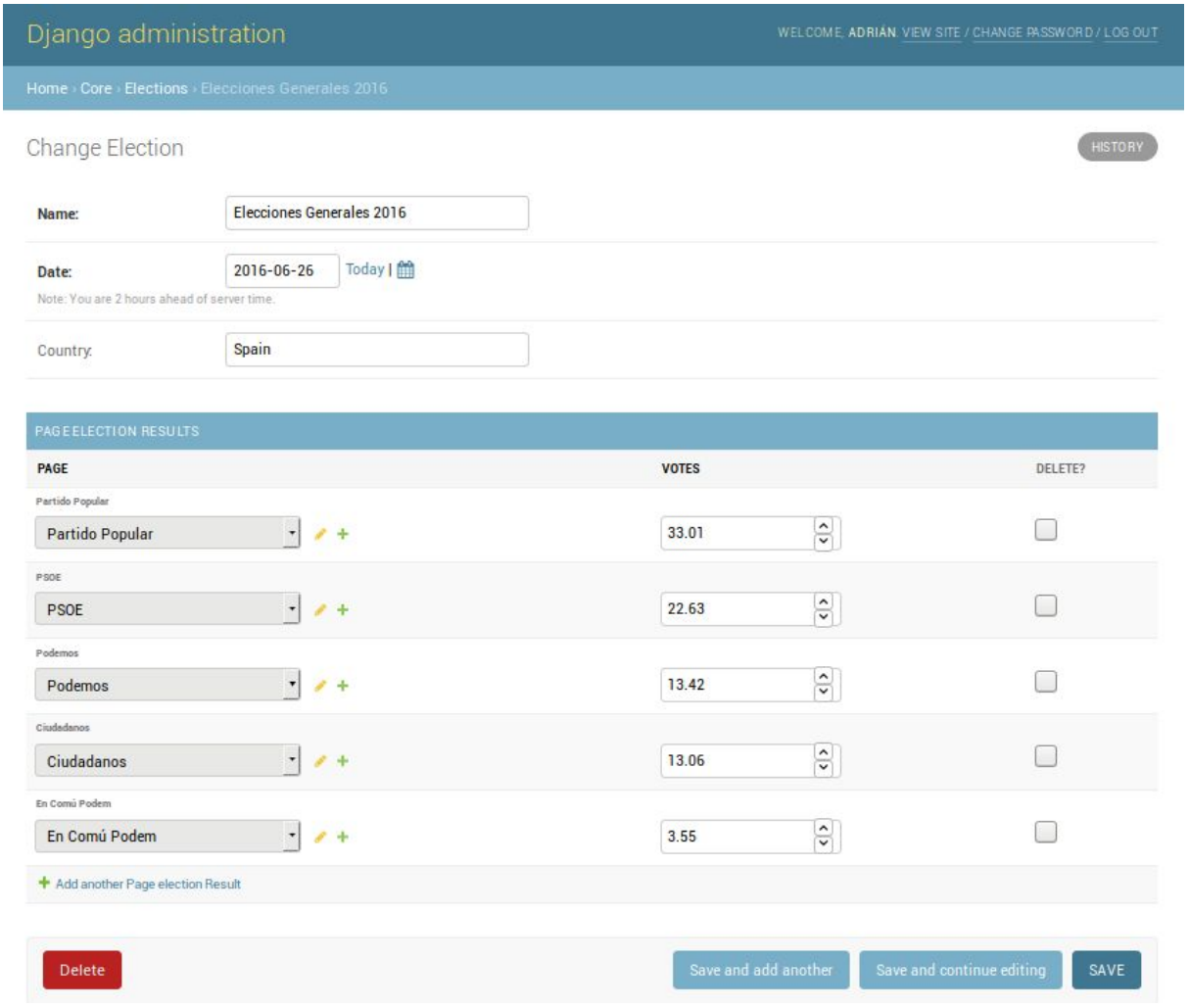
Previo a la extracción de datos se ha realizado una selección de elecciones con resultados históricos, concretamente enfocados al caso de España. Tal es así que las elecciones que se han tenido en cuenta han sido las elecciones generales de 2015 (20 de diciembre de 2015) y las elecciones generales de 2016 (26 de junio de 2016).

Se ha tenido en cuenta que estos procesos electorales fueran realizados en un periodo relativamente moderno, de modo que puedan existir suficientes datos en Facebook sobre cada partido. Con todo ello, el periodo que se ha decido utilizar es el comprendido desde el día 1 de diciembre de 2014 hasta el presente.

El histórico de resultados ha sido almacenado también en base de datos usando la siguiente tabla o modelo, usando relaciones hacia el modelo *Page*, ya existente:



Y desde el panel de administración de Django podríamos visualizar el histórico tal como se muestra en la siguiente captura de pantalla:



Indicar que la unidad para el campo de votos es porcentaje.

6.3. Extracción

Información sobre páginas

Llegados a este punto se procede a realizar los pasos mencionados al comienzo del capítulo, para lo cual, una vez identificadas las páginas de Facebook, se utiliza el identificador de cada una de ellas para añadirla a nuestra base de datos haciendo uso del siguiente comando:

```
$ python manage.py extract --pages=<FACEBOOK-ID> --token=<TOKEN>
```

O haciéndolo usando varios identificadores en el mismo comando:

```
$ python manage.py extract --pages=<FACEBOOK-ID1>,...,<FACEBOOK-IDn>
                           --token=<TOKEN>
```

Información sobre publicaciones

Una vez tenemos todas las páginas en nuestro sistema se procede a extraer las publicaciones para un rango de fechas determinado. En este caso no hace falta especificar cada página por separado pues ahora, teniendo los datos de cada página, se puede filtrar por país de la siguiente forma:

```
$ python manage.py extract --country=<PAIS> --token=<TOKEN> --get_posts
            --start_data=2014-12-01 --end_date=2018-06-15
```

Información sobre comentarios

Por último, para la extracción de comentarios de cada publicación simplemente modificamos el comando anterior, usando el parámetro *--get_comments* como se aprecia a continuación:

```
$ python manage.py extract --country=<PAIS> --token=<TOKEN> --get_comments
            --start_data=2014-12-01 --end_date=2018-06-15
```

El sistema de extracción no usa ningún tipo de multiprocesamiento (threads o hilos) pues al incrementar la capacidad de extracción, disminuyendo el tiempo, se podría incurrir en alcanzar los límites de la API de Facebook. De este modo aunque sea un proceso que pueda llevar más tiempo, es más seguro al no tener que hacer uso de distintos tokens de usuario.

7. Predicción

Para esta labor haremos uso de scikit-learn, una librería para Python para trabajar con Machine Learning[22].

Machine learning es un subconjunto de la inteligencia artificial, que usa técnicas para proveer la habilidad de "aprender" con dato, haciendo uso por ejemplo de mejora progresiva en tareas específicas, sin haber sido programado explícitamente para ello.

La librería scikit-learn provee de herramientas eficientes y simples para minería de datos y análisis. Está programada usando Numpy, SciPy y Matplotlib. Además, es open source, bajo licencia BSD[23], lo que hace que sea ampliamente usada incluso para desarrollos comerciales.

Esta librería está disponible en pip (python package index), que al igual que el resto de librerías usadas para este trabajo, ha sido instalada haciendo uso del comando:

```
$ pip install scikit-learn
```

Con scikit-learn disponible en nuestro entorno, podemos comenzar a utilizarlo.

7.1. Problema de aprendizaje

En general, un problema de aprendizaje considera un conjunto de N muestras de datos y luego trata de predecir propiedades de datos desconocidos. Si cada muestra es más de un número único y, por ejemplo, una entrada multidimensional (también conocida como datos multivariantes), se dice que tiene varios atributos o características.

Podemos separar los problemas de aprendizaje en algunas categorías grandes: aprendizaje supervisado y no supervisado.

[22] Aprendizaje automático (Wikipedia): https://es.wikipedia.org/wiki/Aprendizaje_autom%C3%A1tico
[23] Licencia BSD (Wikipedia): https://en.wikipedia.org/wiki/BSD_licenses#4-clause_license_(original_%22BSD_License%22)

Aprendizaje supervisado

En el cual los datos vienen con atributos adicionales que queremos predecir. Este problema puede ser: clasificación o regresión.

Clasificación

Las muestras pertenecen a dos o más clases y queremos aprender de los datos ya etiquetados cómo predecir la clase de datos sin etiqueta. Un ejemplo de problema de clasificación sería el ejemplo de reconocimiento de dígitos escritos a mano, en el que el objetivo es asignar cada vector de entrada a uno de un número finito de categorías discretas. Otra forma de pensar en la clasificación es como una forma discreta (en lugar de continua) de aprendizaje supervisado donde uno tiene un número limitado de categorías y para cada una de las N muestras proporcionadas, una es tratar de etiquetarlas con la categoría o clase correcta.

Regresión

Si el resultado deseado consiste en una o más variables continuas, entonces la tarea se llama regresión. Un ejemplo de un problema de regresión sería la predicción de la longitud de un salmón en función de su edad y peso. Nosotros usaremos este método.

Aprendizaje no supervisado

Los datos de entrenamiento consisten en un conjunto de vectores de entrada X sin ningún valor objetivo correspondiente.

El objetivo en tales problemas puede ser descubrir grupos de ejemplos similares dentro de los datos, donde se llama agrupamiento, o determinar la distribución de datos dentro del espacio de entrada.

7.2. Estrategia

La estrategia que se ha seguido consiste en obtener los datos inmediatamente pasados y próximos a cada uno de los procesos electorales, por ejemplo de los 45 días previos a cada una de las elecciones y agruparlos. Estos datos pueden ser escogidos en función de la relevancia que puedan tener, realizando además pruebas que nos permitan decidir cuál sería el mejor conjunto de datos a analizar.

Por ejemplo, en el hipotético caso de querer usar solo el valor del sentimiento de Watson y tomando 45 días previo a cada proceso electoral como periodo de

observación, para el caso de España tendríamos los siguientes datos (la lista se muestra por orden alfabético):

Ciudadanos

Elecciones	Rango observación	Media del sentimiento (Watson)	% de votos obtenidos
Generales 2015 (20/11/2015)	05/11/2015 - 20/11/2015	0.64283946875	13.94
Generales 2016 (26/06/2016)	12/05/2016 - 26/06/2016	0.38609353211009173	13.06

En Comú Podem

Elecciones	Rango observación	Media del sentimiento (Watson)	% de votos obtenidos
Generales 2015 (20/11/2015)	05/11/2015 - 20/11/2015	0.3888812710280375	3.69
Generales 2016 (26/06/2016)	12/05/2016 - 26/06/2016	0.36519425974025965	3.55

PSOE

Elecciones	Rango observación	Media del sentimiento (Watson)	% de votos obtenidos
Generales 2015 (20/11/2015)	05/11/2015 - 20/11/2015	0.2917352578125	22.0
Generales 2016 (26/06/2016)	12/05/2016 - 26/06/2016	0.28704537654320994	22.63

Partido Popular

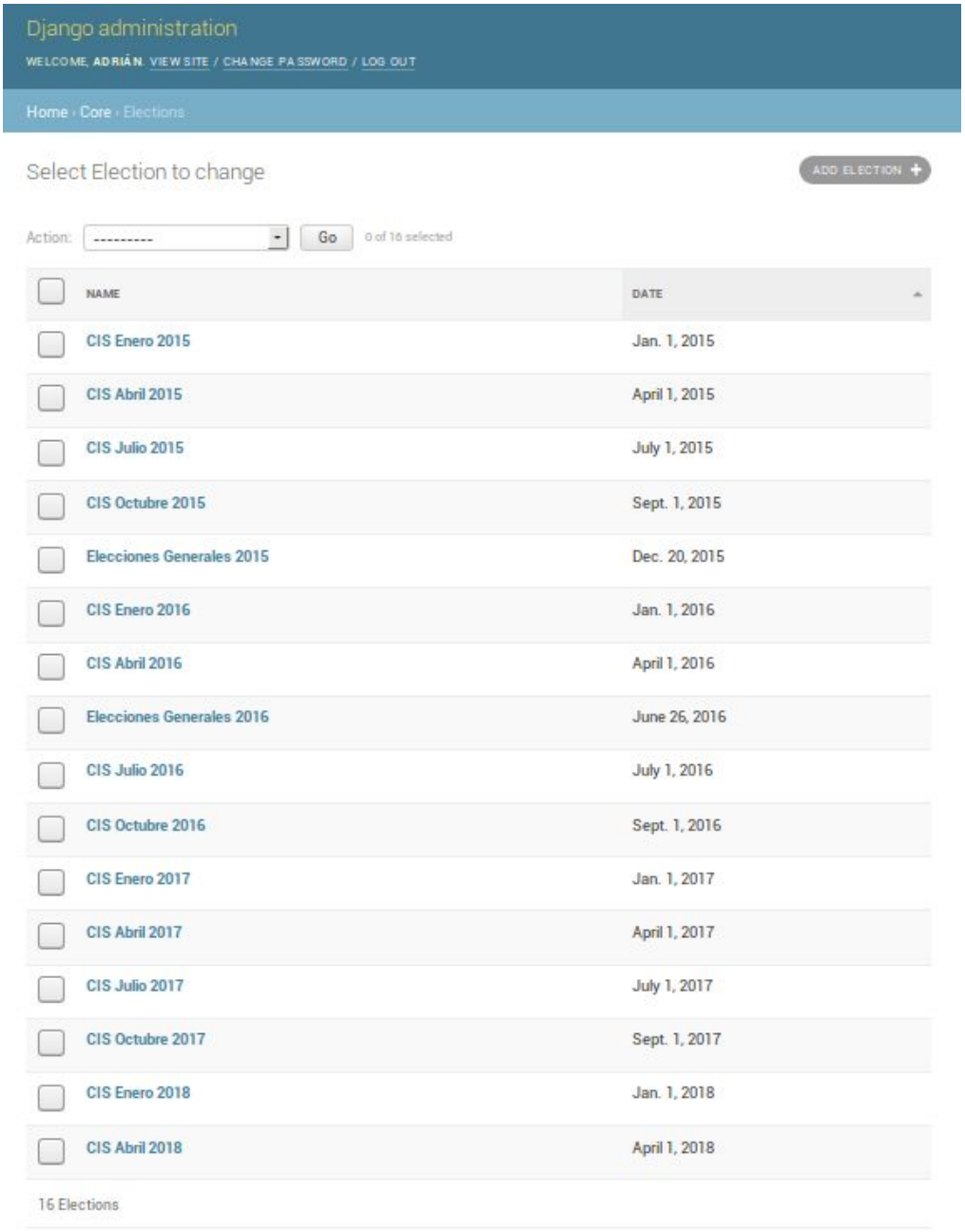
Elecciones	Rango observación	Media del sentimiento (Watson)	% de votos obtenidos
Generales 2015 (20/11/2015)	05/11/2015 - 20/11/2015	0.07658332270916339	28.71
Generales 2016 (26/06/2016)	12/05/2016 - 26/06/2016	0.23345432867132873	33.01

Podemos

Elecciones	Rango observación	Media del sentimiento (Watson)	% de votos obtenidos
Generales 2015 (20/11/2015)	05/11/2015 - 20/11/2015	0.2582649381443299	12.69
Generales 2016 (26/06/2016)	12/05/2016 - 26/06/2016	0.3396056106194689	13.42

A priori, con dos datos que se muestran anteriormente, podemos observar que no es suficiente para entrenar de forma eficiente nuestro sistema de predicción por lo que se decidió añadir los datos de los barómetros del CIS que contienen estimación de voto desde el 1 de Enero de 2015, de este modo contamos con un total de 16 datos de porcentaje de votos, tal como se aprecia en el panel de administración de Django para el modelo de datos Elecciones.

Estos datos han sido extraídos de la página que el CIS pone a disposición para su descarga en el enlace a continuación:
http://www.cis.es/cis/export/sites/default/-Archivos/Indicadores/documentos_html/sB606050020.html



Tal que así, si observamos el mismo ejemplo anterior, para el caso del primer partido de la lista, el número de datos que podremos utilizar para entrar nuestro sistema será mayor:

Tabla: datos históricos de Ciudadanos

Elecciones / Estimación barómetro del CIS	Rango observación	Media del sentimiento (Watson)	% de votos obtenidos
Barómetro del CIS (01/01/2015)	2014-11-17 - 2015-01-01	0.5589013414634146	3.1
Barómetro del CIS (2015-04-01)	2015-02-15 - 2015-04-01	0.44836454545454557	13.8

Barómetro del CIS (2015-07-01)	2015-05-17 - 2015-07-01	0.5217204999999999	11.1
Barómetro del CIS (2015-09-01)	2015-07-18 - 2015-09-01	0.3984549464285715	14.7
Generales 2015 (20/11/2015)	05/11/2015 - 20/11/2015	0.64283946875	13.94
Barómetro del CIS (2016-01-01)	2015-11-17 - 2016-01-01	0.6743260512820515	13.3
Barómetro del CIS (2016-04-01)	2016-02-16 - 2016-04-01	0.2641133571428571	15.6
Generales 2016 (26/06/2016)	12/05/2016 - 26/06/2016	0.38609353211009173	13.06
Barómetro del CIS (2016-07-01)	2016-05-17 - 2016-07-01	0.3818161964285714	12.0
Barómetro del CIS (2016-09-01)	2016-07-18 - 2016-09-01	0.13230981034482758	12.8
Barómetro del CIS (2017-01-01)	2016-11-17 - 2017-01-01	0.17740554000000003	12.4
Barómetro del CIS (2017-04-01)	2017-02-15 - 2017-04-01	0.07102822471910114	14.9
Barómetro del CIS (2017-07-01)	2017-05-17 - 2017-07-01	0.0944789090909091	14.5
Barómetro del CIS (2017-09-01)	2017-07-18 - 2017-09-01	0.09374849019607845	17.5
Barómetro del CIS (2018-01-01)	2017-11-17 - 2018-01-01	0.3331265641025641	20.7
Barómetro del CIS (2018-04-01)	2018-02-15 - 2018-04-01	0.2428558315789473	22.4

Es importante comentar, que aunque el número de muestras puede ser suficiente, el número de características es inadecuado. Por ello, en este punto decidimos introducir los siguientes valores para cada muestra:

- Número de publicaciones para el rango de fechas
- Número de comentarios de las publicaciones
- Número de veces que se han compartido las publicaciones

- Número de reacciones por separado: likes, loves, hahas, wows, sads y angrys

Poniendo todo esto en común, para cada rango de observación se tendrán más datos y hará que nuestro sistema pueda ser entrenado de una forma más eficiente. Pongamos el ejemplo de la última muestra, para el barómetro del CIS con fecha 1 de abril de 2018 y por tanto el rango de 45 días previo, es decir entre el 15 de febrero de 2018 y la propia fecha del barómetro los datos con los que contamos son:

```
[97, 15675, 46571, 116025, 11259, 3099, 473, 1774, 6090, 0.2428558315789473] = 22.4
```

Representado en formato de array, y por orden sería: publicaciones, comentarios, veces compartidos, likes, loves, hahas, wows, sads, angrys y valor de sentimiento (watson). A esta parte izquierda se la conoce también como *features*. En el lado derecho se encuentra el valor del porcentaje de votos, también llamado *target*.

Ahora podemos decir que el partido que generó esta interacción y obtuvo esa média en el análisis de sentimiento (valores de la izquierda) ha obtenido el número de porcentaje de votos que están a la derecha.

7.3. Modelo de aprendizaje

Con scikit-learn tenemos a disposición distintos modelos de aprendizaje. Tal como comentamos anteriormente usaremos un modelo de regresión. De todos los modelos que se han realizado pruebas el que mejor resultados ha proporcionado es el DecisionTreeRegressor[24].

```
model = DecisionTreeRegressor': DecisionTreeRegressor(
        random_state=numpy.random.RandomState(0)
)
```

7.4.Entrenamiento del modelo

Para llevar a cabo del entrenamiento se procede a usar los datos de resultados históricos (ver ejemplo en la página anterior). De forma que almacenamos en training_data los distintos *features* y en training_target el *target*.

[24] http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

A continuación se muestra una porción de código usado para el entrenamiento:

```
training_data.append(row)
training_target.append(election_page.votes)
X = training_data
y = training_target
model.fit(training_data, training_target)
```

Ahora el modelo está listo para realizar predicciones, solo hay que usar la función *predict()* como se muestra en las siguientes líneas:

```
Xnew = [observation_data]
ynew = model_func.predict(Xnew)
```

Se puede apreciar cómo se realiza una predicción para un nuevo conjunto de datos, almacenados en *Xnew*, y como se guarda el valor de porcentaje de votos en la variable *ynew*.

8. Análisis y Resultados

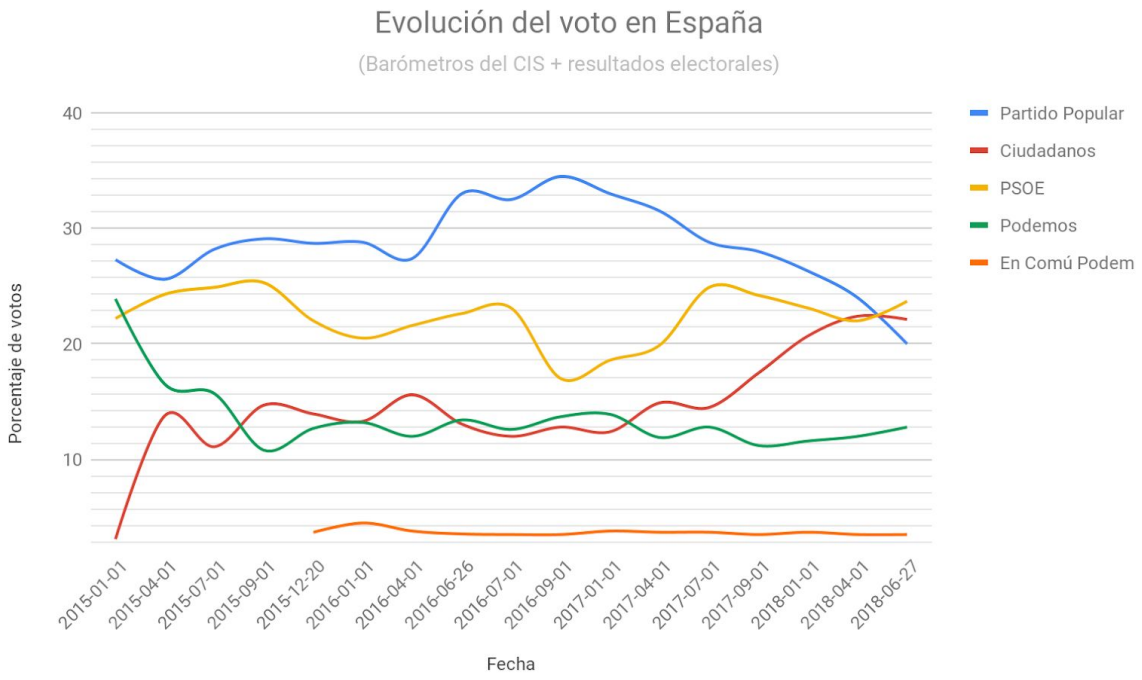
A continuación se muestra la evolución del voto en España para los barómetros del CIS así como los resultados reales de los dos procesos electorales del 20 de diciembre de 2015 y del 26 de junio de 2016.

	Porcentaje de votos				
Fecha	Partido Popular	Ciudadanos	PSOE	Podemos	En Comú Podem
2015-01-01	27.3	3.1	22.2	23.9	
2015-04-01	25.6	13.8	24.3	16.5	
2015-07-01	28.2	11.1	24.9	15.7	
2015-09-01	29.1	14.7	25.3	10.8	
2015-12-20	28.71	13.94	22	12.69	3.69
2016-01-01	28.8	13.3	20.5	13.2	4.5
2016-04-01	27.4	15.6	21.6	12	3.8
2016-06-26	33.01	13.06	22.63	13.42	3.55
2016-07-01	32.5	12	23.1	12.6	3.5
2016-09-01	34.5	12.8	17	13.7	3.5
2017-01-01	33	12.4	18.6	13.9	3.8
2017-04-01	31.5	14.9	19.9	11.9	3.7
2017-07-01	28.8	14.5	24.9	12.8	3.7
2017-09-01	28	17.5	24.2	11.2	3.5
2018-01-01	26.3	20.7	23.1	11.6	3.7

Estos datos se encuentran en la base de datos, después de haber realizado la recopilación de datos procedentes de la web del CIS: http://www.cis.es/cis/export/sites/default/-Archivos/Indicadores/documentos_html/sB606050020.html

Así como de la web del Ministerio del interior para consulta de procesos electorales: http://www.infoelectoral.mir.es/infoelectoral/min/busquedaAvanzadaAction.html

También se muestra el gráfico de los datos anteriores para una mejor interpretación de los mismos, donde se pueda apreciar cambios y tendencias.



8.1. Test del modelo de predicción

Para poner a prueba nuestro modelo de predicción podemos realizar predicciones para fechas en el rango del gráfico anterior para los que no existen valores, por ejemplo podemos escoger el 1 de Mayo de 2016, para el que no tenemos valor de porcentaje de votos y aplicarlo a nuestro modelo para comprobar la salida.

Además tenemos una variable que podemos usar para ajustar el modelo de predicción, esta es días de observación, representada en nuestro software con el parámetro *--observation_days*. Esta variable puede ser un valor numérico o un rango, de tal modo que podríamos pasar [N_1, N_2] indicando que queremos realizar la predicción tomando valores entre N_1 y N_2 de forma iterativa. Al finalizar se muestra la media aritmética del valor para la predicción.

Veamos el ejemplo con un test real, ejecutando el siguiente comando:

```
python manage.py predict --predict_date=2016-05-01 --observation_days=[20,25] --country=Spain
```

Los parámetros usados son:

- --prediction_data: indica la fecha para la cual queremos tener el valor estimado.
- --observation_days: que cantidad de días se usará para recopilar datos y entrenar el model. En caso de usar un rango se realiza varias iteraciones, por lo que se obtienen varias salidas con distintas predicciones.
- --country: usado para filtrar por país y realizar el estudio para los partidos políticos de un país concreto.

A continuación se muestra el resultado de la ejecución de cada partido.

Ciudadanos

Días obs.	Publicaciones	Comentarios	Veces comp.	Likes	Love	Haha	Wow	Sad	Angry	Sent. Watson	Est. voto
20	40	7137	19879	54549	5454	1209	192	414	2509	0.2175	15.6
21	43	7444	20876	57344	5597	1217	195	965	2561	0.2031	14.9
22	47	7923	21411	59393	5981	1231	196	965	2568	0.2277	14.7
23	48	8017	21591	60088	6035	1259	197	965	2574	0.2228	14.7
24	51	8623	23239	63122	6177	1326	239	1243	3218	0.2344	14.7
25	53	8731	23856	64305	6241	1343	250	1248	3291	0.2252	14.5
										Media aritmética	14.85

En Comú Podem

Días obs.	Publicaciones	Comentarios	Veces comp.	Likes	Love	Haha	Wow	Sad	Angry	Sent. Watson	Est. voto
20	44	88	410	869	48	8	2	14	64	0.4417	3.7
21	45	98	414	872	48	8	2	14	80	0.4494	3.7
22	48	98	422	886	48	8	2	19	86	0.458	3.7
23	51	99	446	959	57	8	2	19	86	0.4653	3.7
24	53	107	457	1057	67	8	2	19	86	0.4653	3.7
25	58	111	497	1164	68	8	2	19	87	0.4719	3.7
										Media aritmética	3.7

PSOE

Días obs.	Publicaciones	Comentarios	Veces comp.	Likes	Love	Haha	Wow	Sad	Angry	Sent. Watson	Est. voto
20	54	10223	11068	19082	2353	762	43	91	736	0.0829	23.1
21	56	10379	11720	19755	2367	762	46	354	754	0.0602	23.1
22	58	10680	12004	20253	2469	778	46	354	758	0.0585	23.1
23	61	10919	12279	20875	2580	790	49	358	766	0.0471	23.1
24	66	11040	12917	22495	2728	814	52	362	767	0.063	20.5
25	70	11097	13262	23163	2758	828	52	410	769	0.0718	17
										Media aritmética	21.65

Partido Popular

Días obs.	Publicaciones	Comentarios	Veces comp.	Likes	Love	Haha	Wow	Sad	Angry	Sent. Watson	Est. voto
20	72	7575	12870	22227	2548	507	109	623	1770	0.1979	27.3
21	75	8007	13591	23103	2686	519	112	788	1784	0.2056	27.3
22	78	8260	13812	23700	2786	553	124	789	1824	0.2141	34.5
23	82	8479	14180	24358	2853	578	126	789	1838	0.2259	34.5
24	91	9096	14965	25844	3085	609	130	789	1858	0.2389	27.3
25	95	9598	15899	27696	3293	624	149	796	1988	0.2383	28
										Media aritmética	29.82

Podemos

Días obs.	Publicaciones	Comentarios	Veces comp.	Likes	Love	Haha	Wow	Sad	Angry	Sent. Watson	Est. voto
20	69	24767	128939	105785	14918	9298	2062	2543	14393	0.258	12.8
21	72	25745	144505	118253	17301	9345	2098	2655	14477	0.257	12
22	76	27642	147376	122654	18572	9408	2108	2663	14559	0.2614	12
23	78	27855	152897	126384	18941	9428	2131	2701	15452	0.2366	12
24	81	28530	174497	142651	24631	9493	2360	2708	15461	0.2461	12
25	91	37675	185385	158343	40206	9671	2394	2811	16487	0.2371	12
										Media aritmética	12.13

Como se aprecia en los datos de valor estimado de votos, la predicción varía en función del número de días de observación que se tome, haciendo que el valor final se pueda ver afectado. Para evitar esto al finalizar el cálculo para el rango dado de días se realiza la media aritmética haciendo así que la predicción que vamos a tomar tenga un error menor.

Para la fecha que hemos utilizado podemos observar que el valor calculado de estimación de votos se encuentra en sintonía con los valores de la tabla de estimación de voto al comienzo de este capítulo.

8.2. Resultados

Una vez validado nuestro modelo procedemos a realizar predicciones en el futuro.

Aprovechando que a la finalización de este trabajo el barómetro del CIS del mes de julio no ha sido aún publicado, podemos hacer una predicción con fecha próxima al 1 de julio y analizar los resultados. La razón de no poder hacer una predicción con fecha 1 de julio es debido a que para hacerlo necesitaríamos datos de Facebook para al menos esa fecha, pues esos datos son usados para entrenar nuestro modelo de predicción. Por todo ello, la predicción se realiza a fecha 25 de junio de 2018.

A continuación se muestra el resultado de la ejecución de cada partido.

```
python manage.py predict --predict_date=2018-06-25 --observation_days=[20,25] --country=Spain
```

Los resultados para cada partido en formato de tabla, después de ejecutar el script:

Ciudadanos

Días obs.	Publicaciones	Comentarios	Veces comp.	Likes	Love	Haha	Wow	Sad	Angry	Sent. Watson	Est. voto
20	40	7137	19879	54549	5454	1209	192	414	2509	0.22	12
21	43	7444	20876	57344	5597	1217	195	965	2561	0.2	12
22	47	7923	21411	59393	5981	1231	196	965	2568	0.23	22.4
23	48	8017	21591	60088	6035	1259	197	965	2574	0.22	22.4
24	51	8623	23239	63122	6177	1326	239	1243	3218	0.23	22.4
25	53	8731	23856	64305	6241	1343	250	1248	3291	0.23	22.4
										Media aritmética	18.93

En Comú Podem

Días obs.	Publicaciones	Comentarios	Veces comp.	Likes	Love	Haha	Wow	Sad	Angry	Sent. Watson	Est. voto
20	44	88	410	869	48	8	2	14	64	0.44	3.5
21	45	98	414	872	48	8	2	14	80	0.45	3.55
22	48	98	422	886	48	8	2	19	86	0.46	3.5
23	51	99	446	959	57	8	2	19	86	0.47	3.5
24	53	107	457	1057	67	8	2	19	86	0.47	3.55
25	58	111	497	1164	68	8	2	19	87	0.47	3.5
										Media aritmética	3.52

PSOE

Días obs.	Publicaciones	Comentarios	Veces comp.	Likes	Love	Haha	Wow	Sad	Angry	Sent. Watson	Est. voto
20	54	10223	11068	19082	2353	762	43	91	736	0.08	24.9
21	56	10379	11720	19755	2367	762	46	354	754	0.06	24.9
22	58	10680	12004	20253	2469	778	46	354	758	0.06	24.9
23	61	10919	12279	20875	2580	790	49	358	766	0.05	23.1
24	66	11040	12917	22495	2728	814	52	362	767	0.06	20.5
25	70	11097	13262	23163	2758	828	52	410	769	0.07	20.5
										Media aritmética	23.14

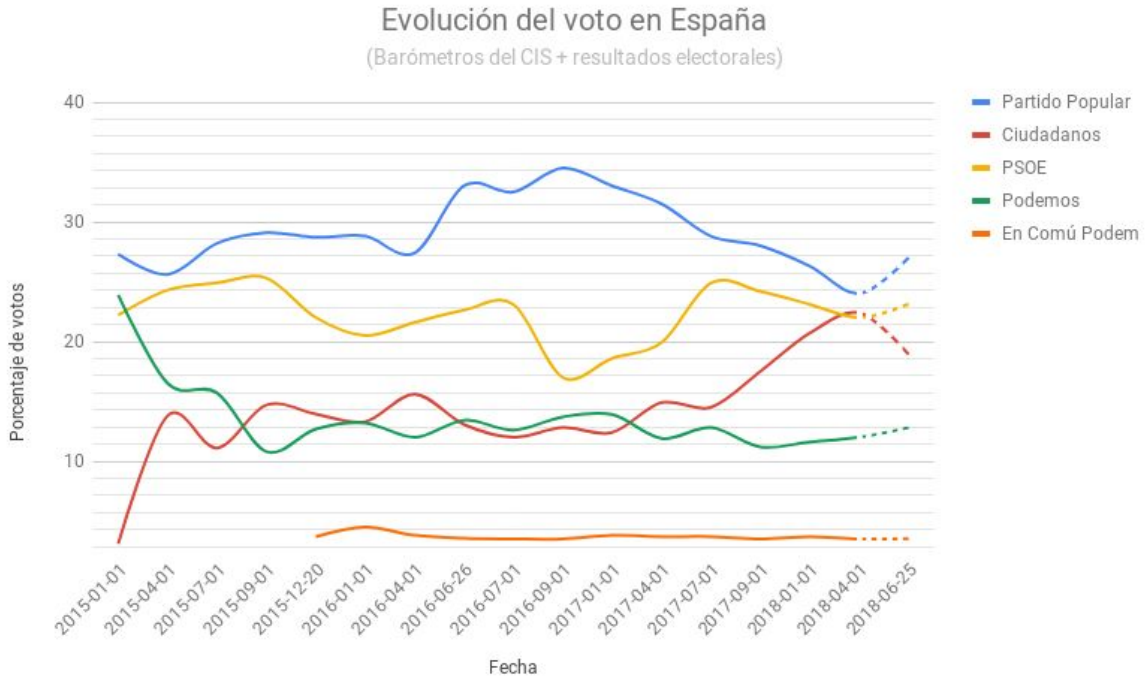
Partido Popular

Días obs.	Publicaciones	Comentarios	Veces comp.	Likes	Love	Haha	Wow	Sad	Angry	Sent. Watson	Est. voto
20	72	7575	12870	22227	2548	507	109	623	1770	0.2	24
21	75	8007	13591	23103	2686	519	112	788	1784	0.21	24
22	78	8260	13812	23700	2786	553	124	789	1824	0.21	24
23	82	8479	14180	24358	2853	578	126	789	1838	0.23	24
24	91	9096	14965	25844	3085	609	130	789	1858	0.24	33.01
25	95	9598	15899	27696	3293	624	149	796	1988	0.24	33.01
										Media aritmética	27

Podemos

Días obs.	Publicaciones	Comentarios	Veces comp.	Likes	Love	Haha	Wow	Sad	Angry	Sent. Watson	Est. voto
20	69	24767	128939	105785	14918	9298	2062	2543	14393	0.26	13.2
21	72	25745	144505	118253	17301	9345	2098	2655	14477	0.26	13.2
22	76	27642	147376	122654	18572	9408	2108	2663	14559	0.26	13.2
23	78	27855	152897	126384	18941	9428	2131	2701	15452	0.24	13.2
24	81	28530	174497	142651	24631	9493	2360	2708	15461	0.25	12
25	91	37675	185385	158343	40206	9671	2394	2811	16487	0.24	12
										Media aritmética	12.8

Y para facilitar la representación de resultados podemos observar el resultado de forma lineal.



8.3. Conclusiones

En este trabajo de fin de Máster se ha realizado una aplicación web en Django donde hemos recopilado datos de Facebook usando su API y los hemos almacenado en base de datos para poder realizar un tratamiento posteriormente.

Una vez obtenidos los datos se han usados dos herramientas para el análisis de sentimientos, aunque finalmente solo usamos uno para el cálculo final (Watson, de IBM).

Posteriormente, mediante el uso de la librería de Python, scikit-learn, hemos mostrado cómo realizar un modelo con el que obtener predicciones en base al histórico de datos electorales y otros datos procedentes del barómetro del CIS.

Al finalizar este trabajo contamos con una herramienta para predecir resultados electorales, basándonos únicamente en datos de la red social Facebook. Se trata de un sistema que puede ser mejorado para obtener mejores resultados y más certeros.

8.4. Limitaciones

Entre las limitaciones que hemos encontrado se encuentra la API de Facebook, pues no contamos con información detallada de usuarios como ocurría en el

pasado, y poder realizar comportamiento de usuarios para una mayor clasificación de datos.

Por otro lado, la herramienta de IBM para análisis de sentimiento, Watson, es un sistema que funciona con suscripción y por lo tanto tiene un coste económico por lo que sólo se pudo hacer uso de la cuenta demo que IBM provee, no teniendo la libertad en cuenta a cantidad de datos a analizar que quizás hubiéramos necesitado.

En la fecha en la que se realizó las predicciones finales para este trabajo, existían primarias en uno de los partido políticos, lo que produjo que en dicho partido existiera una sobre-interacción en comparación con otros partido, pudiendo afectar al resultado de predicción.

8.5. Mejoras del sistema

Como mejoras principales para futuros desarrollos/investigaciones sobre temas políticos, sería posible trabajar en los siguientes puntos:

Histórico de Fans de cada página de Facebook

Facebook no provee del histórico de fans para cad página, por ello sería necesario realizar la extracción a diario con el fin de obtener el dato de crecimiento de seguidores. Con este dato se podría realizar un mejor entrenamiento que ayude como una mejor predicción.

Nuevas fuentes de datos

Una idea podría ser incorporar datos de otras redes sociales como Twitter, y realizar búsquedas por hashtags.

Además otra idea sería tener en cuenta procesos judiciales que puedan salpicar a partidos políticos y las imputaciones en casos de corrupción para cada partido, analizando cómo puede afectar esto a cada partido.

Bibliografía

Documentación sobre los barómetros y encuestas pre-electorales del Centro de Investigaciones Sociológicas:

- http://www.cis.es/cis/opencms/ES/index.html

Información de histórico de procesos electorales en España:

- http://www.infoelectoral.mir.es/infoelectoral/min/busquedaAvanzadaAction.html

Documentación detallada relacionada con la API de Facebook:

- https://developers.facebook.com/docs/graph-api/overview/
- https://developers.facebook.com/docs/graph-api/reference/page
- https://developers.facebook.com/docs/graph-api/reference/page/feed
- https://developers.facebook.com/docs/graph-api/reference/post
- https://developers.facebook.com/docs/graph-api/reference/object/comments

Interfaces de Aplicación Programables (API):

- https://es.wikipedia.org/wiki/Interfaz_de_programaci%C3%B3n_de_aplicaciones

Python SDK para Facebook Graph API:

- https://github.com/mobolic/facebook-sdk

Herramientas de análisis de sentimiento:

- IBM:
    - http://bluemix-watson-day.mybluemix.net/
    - https://natural-language-understanding-demo.ng.bluemix.net/
    - https://www.ibm.com/watson/developercloud/natural-language-understanding/api/v1/
    - https://console.bluemix.net/docs/services/natural-language-understanding/getting-started.html#getting-started-tutorial

- NLT (Natural Language Toolkit):
    - https://www.nltk.org/

Scikit-learn (librería para machine Learning con Python):

- http://scikit-learn.org/stable/
- http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html