



OCTOPUS

DATA INSIGHTS

ANÁLISIS DE REDES SOCIALES PARA RECURSOS HUMANOS

Fabio Inui

Teresa Martínez

Javier Quintana

Silvia Santos

Dedicamos esta memoria a nuestras familias, sin cuya paciencia sin límites, no hubiera sido posible su elaboración.

Agradecemos a nuestros profesores su apoyo, y en especial a nuestra tutora Mariluz Congosto, la amable ayuda que nos ha prestado siempre que hemos recurrido a ella.

Contenidos

Resumen ejecutivo	3
Executive summary	5
1 Planteamiento del proyecto	7
1.1 Descripción	7
1.2 Contexto de negocio	10
1.3 Objetivos y criterio de éxito	15
1.4 Hipótesis y limitaciones	15
1.4.1 La Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal	16
1.5 Esquema de desarrollo del proyecto	20
2 Planificación del proyecto	23
2.1 Equipo	23
2.2 Desarrollo temporal	23
3 Infraestructura	27
3.1 Repositorio GIT	27
3.2 Infraestructura para la obtención de datos	27
3.3 Lenguajes de implementación	30
3.4 Almacenamiento y manipulación de los datos	30
3.5 Fuentes de información	31
4 Obtención y tratamiento inicial de los datos: análisis descriptivo	33
4.1 Descripción de los datos	33
4.2 Obtención de los datos	36
4.3 Almacenamiento	37
4.4 Revisión inicial de los datos	37
5 Limpieza de datos: extracción de usuarios relevantes	49
5.1 Detección del idioma	50

5.2	Naturaleza del tuit	53
5.2.1	Construcción del modelo	53
5.2.2	Resultados del modelo de clasificación	58
5.2.3	Resultados de la clasificación del contenido del tuit	59
5.2.4	Listado de usuarios únicos	61
5.2.5	Listado final de usuarios	61
5.3	Tipo de usuario	61
5.3.1	Revisión de la literatura	62
5.3.2	Ánalisis de los datos para la construcción del modelo	64
5.3.3	Modelo final	67
5.3.4	Eliminación de duplicados	68
5.3.5	Resultados de la selección	69
6	Modelado de los datos: ordenación de los usuarios	71
6.1	Índice h	72
6.2	Grafo relacional	73
6.2.1	El papel de los usuarios: medidas de centralidad	75
6.2.2	La estructura de la red	79
6.3	Almacenamiento	81
6.4	Resultados	82
6.4.1	Ordenación de los usuarios	82
6.4.2	Propiedades del grafo	83
7	Visualización de los resultados	85
7.1	Herramientas	85
7.2	Descripción de la interfaz	85
7.2.1	La pestaña con la lista de usuarios ordenados: “Listado de candidatos” . . .	86
7.2.2	La estructura de la red: pestaña “Grafo”	89
8	Áreas de mejora	93
8.1	Mejoras en el algoritmo de clasificación del contenido del tuit	94
Bibliografía		97

Resumen ejecutivo



El problema

Encontrar el mejor candidato para una oferta de trabajo en un entorno cada vez más digitalizado. ¿Es posible usar la información de las redes sociales para optimizar el proceso?

Una nueva fuente de información: Twitter

Usaremos los tuits publicados con contenido relativo a la oferta de trabajo, y luego usaremos nuestros algoritmos innovadores para:

- descubrir en qué lenguajes tuitean los usuarios
- descubrir si los tuits seleccionados son relevantes, y de aquí extraer los usuarios activos en ese perfil
- descubrir qué usuarios son personas candidatas a la oferta



¿Qué candidatos son los mejores?



Usaremos dos tipos de algoritmos de ordenación para obtener la lista de los mejores candidatos:

- Por contenido publicado: índice h
- Por relevancia en la red de usuarios: medidas de centralidad (por grado, por autovector, Bonacich, PageRank, cercanía e intermediación)

¡Mira en Shiny la lista ordenada de los usuarios!

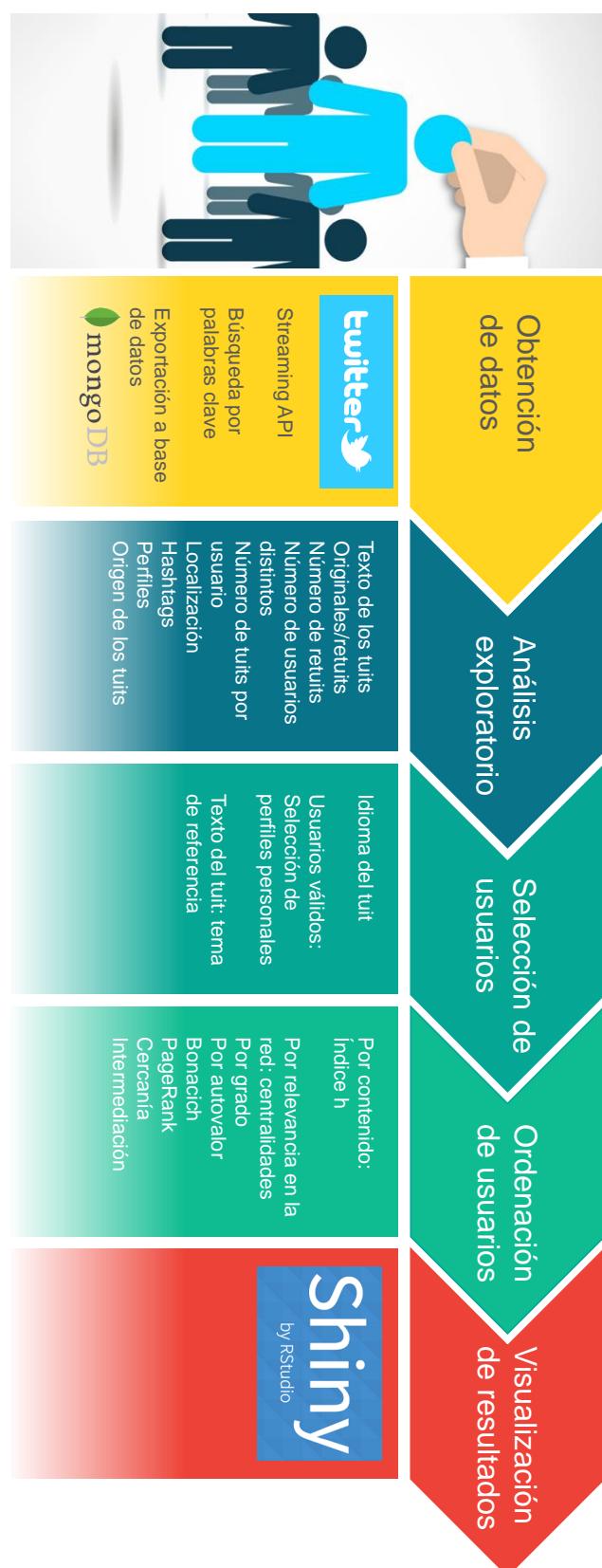


Figura 0.0.1: Flujo de trabajo esquemático.

Executive summary



The problem

You need to find the best candidate for a job offer, in a increasingly digitized environment. Could you use the information contained in social networks to optimize the process?

A new information source: Twitter

We will retrieve published tweets with content related to the job offer, and then use our innovative algorithms to:

- identify tweets languages,
- select the tweets by the relevance of its contents, and extract the active users with the required profile,
- propose the outstanding candidates for the job offer.



Which are the best candidates?

We use two different ordering algorithms to rank the list of the candidates and find the best ones:

- By published contents: *h*-index
- By relevance in the users network: centrality measures (degree, eigenvector, Bonacich, PageRank, closedness, betweenness).

Go to Shiny to navigate the ordered list!



Figura 0.0.2: Schematic workflow.

Capítulo 1

Planteamiento del proyecto

En este capítulo vamos a describir el contexto en el que vamos a desarrollar el contenido del proyecto y las ideas involucradas en su desarrollo.

1.1 Descripción

Cuando un departamento de Recursos Humanos o una empresa de reclutamiento se enfrenta a una petición para cubrir un puesto vacante o de nueva creación, el proceso suele llevarse a cabo en diversas fases, que podríamos describir del siguiente modo [1]:

1. **Preselección:** etapa inicial en la que se detectan candidatos adecuados para el perfil buscado, bien recurriendo a anuncios en portales especializados, bien con búsquedas personalizadas de perfiles. En esta etapa se elabora una lista de candidatos que pasarán a las siguientes fases del proceso, descartando aquellos cuyas competencias no sean las adecuadas para el puesto.
2. **Entrevista inicial:** en esta etapa los candidatos seleccionados en la etapa anterior son contactados para conseguir ampliar la información de la que se dispone sobre ellos (por ejemplo sobre las aptitudes particulares y experiencias previas consignadas en el CV), y verificar el interés y compromiso del candidato con respecto a la oferta.
3. **Informe:** tras la entrevista inicial, se seleccionan los mejores candidatos para el puesto, y se realiza un informe donde se consignan los datos originales (el CV, por ejemplo) y los datos añadidos en el curso de la entrevista inicial.
4. **Presentación de candidatos:** el empleador recibe el informe elaborado en el punto anterior, y selecciona aquellos que mejor se ajusten a sus necesidades, muy habitualmente realizando nuevas entrevistas con ellos.
5. **Decisión:** es el momento en que se elige el candidato al que se le va a ofrecer el puesto, etapa en la que puede complementarse la información recogida hasta el momento con referencias recabadas de anteriores empleadores.

6. **Oferta:** etapa en la que la empresa presenta al candidato la oferta en firme, habitualmente por escrito, consignando la voluntad de la empresa de incorporar al candidato y los detalles económicos.
7. **Seguimiento:** para comprobar que una vez incorporado a la empresa, tanto empleado como empleador están conformes con el resultado del proceso.

Tradicionalmente, el comienzo de este proceso, la detección de candidatos, se realizaba en numerosas ocasiones a través de anuncios en prensa, bases de datos de candidatos construidas a lo largo del tiempo, y la explotación de la red de contactos personales del entorno del empleador. Hoy por hoy, estos métodos tradicionales han sido complementados, y algunos dirían que prácticamente suplantados, por métodos que explotan la información contenida en la web.

Los técnicos de selección se enfrentan a un mundo muy diverso donde tanto la difusión de los posibles puestos como la información sobre los candidatos para los mismos está diseminada en numerosos formatos, teniendo un papel preponderante diversas plataformas o portales web (InfoJobs, Monster, etc.) y redes sociales en general (LinkedIn, Twitter, Facebook, Instagram, etc.). Desde el punto de vista del técnico de selección, las primeras contienen mucha información sobre las aptitudes de los posibles candidatos, sus conocimientos, formación y experiencia, ya que son portales donde los propios usuarios consignan sus currícula vitae, y también sobre su situación laboral actual y expectativas. En el segundo grupo de fuentes, las redes sociales, hay algunas que tienen el carácter específico de las primeras (LinkedIn es el ejemplo más claro), y hay otras en las que se consigna información diversa, llamémoslas de propósito general, tal vez en mayor medida personal que profesional.

El objetivo de nuestro proyecto es complementar el trabajo habitual de un departamento de Recursos Humanos o de un seleccionador de personal en los portales y redes sociales dedicados al mundo laboral, con información laboral extraída de fuentes menos estándar, como son las redes sociales de propósito general. Estas redes son a menudo aprovechadas por los usuarios para difundir mensajes relacionados con su actividad laboral, y una descripción de su actividad en las redes es relevante desde el punto de vista de un reclutador, en la medida que da información del compromiso de la persona con su actividad, su valoración por parte de otros usuarios, su proactividad, etc.

En este trabajo hemos elegido la red social Twitter por diversos motivos: es una red muy dinámica, fácil de usar, rápida y divertida, que involucra cientos de millones de usuarios activos en todo el mundo: 328 millones según la web de la red. El crecimiento del número de usuarios fue casi exponencial entre 2010 y 2014, si bien últimamente la velocidad a la que adquiere nuevos usuarios ha perido intensidad. Es también una red que, desde sus orígenes, ha puesto a disposición de los interesados los mecanismos necesarios para acceder a la información que atesora, con ciertas limitaciones, pero de forma relativamente sencilla.

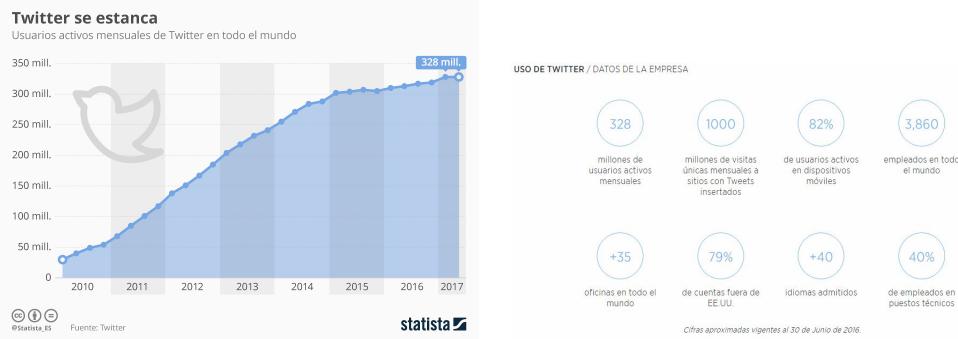


Figura 1.1.1: Twitter: evolución del número de usuarios (Statista 2017, <https://es.statista.com/grafico/10476/el-numero-de-usuarios-de-twitter-se-estanca>) y datos de la empresa, <https://about.twitter.com/es/company>.

Esta red da cabida a relaciones diversas, entre usuarios de variada índole. Dado que muchos de los usuarios publican información relacionada con su ocupación laboral, es natural esperar que en Twitter se formen comunidades de individuos que comparten interés en diferentes aspectos de dicho ámbito. Nuestro propósito es definir e implementar un proceso que permita agregar información referente a esas comunidades a un determinado proceso de selección.

Observemos las dos siguientes ofertas de trabajo aparecidas recientemente (Septiembre 2017) en LinkedIn, incluyendo los requisitos solicitados a los posibles candidatos:

Left Job Post: Data Scientist para equipo BIG DATA
Línea Directa Asesuradora - Madrid y alrededores, España
Publicado hace 3 días · 137 visualizaciones

Right Job Post: Data Scientist
Room Mate Hotels - Madrid y alrededores, España
Publicado hace 2 días · 180 visualizaciones

Figura 1.1.2: Dos ofertas de empleo.

Left Job Post Requirements:

- Perfil Requerido:

 - Licenciado en Matemáticas, Estadística o Ingeniería informática.
 - Conocimientos y experiencia requerida:

 - Experiencia mínima previa de 3 años en desarrollo de modelos estadísticos clásicos y desarrollo/aplicación de modelos de optimización volumen y rentabilidad.
 - Machine learning, Datamining.
 - Map Reduce en Hadoop o similares.
 - Programación JAVA, Python, R.
 - Conocimientos de SQL.
 - Inglés alto.

 - Valorables:

 - Conocimientos actuariales.
 - Software estadístico SAS.

Right Job Post Requirements:

 - Nivel de experiencia: Algo de responsabilidad
 - Sector: Seguros, Servicios y tecnologías de la información, Investigación
 - Tipo de empleo: Jornada completa
 - Funciones laborales: Análisis, Investigación, Tecnología de la información
 - PERFIL REQUERIDO

 - Experiencia de mínimo 2 años como Data Analyst con familiarización de Big Data, análisis y/o data mining.
 - Experiencia práctica en "ciencias de la decisión", Machine learning y mining de datos.
 - Familiarización con los modelos y métodos estadísticos.
 - Experiencia en programación con una o más herramientas analíticas (SQL, R, Python, Matlab, SAS) y herramientas propias de los sistemas de análisis y visualización Big Data (Qlik view/Sense, Tableau, Power BI...)
 - Capacidad de integración de varias fuentes de datos. Conocimiento de herramientas ETL (Power Center/etc.)
 - Experiencia demostrable en Data Warehouse.

 - REQUISITOS MÍNIMOS

 - Titulación superior en Matemáticas, Estadística, Ingeniería o carrera técnica.
 - Alto nivel de inglés. Se valorarán otros idiomas.
 - Capacidad de relación con todos los departamentos internos y externos, proveedores asociados a integraciones y servicios.

 - Tecnología de la información

Figura 1.1.3: Requisitos de las dos ofertas de empleo.

En ambos casos, entre los requisitos se encuentran conocimientos sobre Python, R, SQL, machine learning y data mining. Un reclutador probablemente usará esas palabras clave para buscar

los perfiles adecuados para alguno de los dos puestos, y construirá un conjunto de posibles candidatos (el primer paso en nuestra descripción del proceso de contratación). En esta fase, y gracias a Octopus Data Insights, nuestro reclutador contará con una ayuda extra. Octopus Data Insights le proporcionará una lista de usuarios de Twitter que hayan publicado contenido en el que aparezcan esas palabras clave, que complementarán el resultado que el reclutador haya obtenido por sus propios medios. La información proporcionada por Octopus Data Insights resultará relevante también más adelante en el proceso, cuando haya que tomar una decisión entre varios candidatos para determinar cuáles son los más adecuados para el puesto: usando la información de Twitter, los usuarios de la lista estarán ordenados según diversos criterios de relevancia.

El proceso para producir la información que ayudará al reclutador en el proceso es el siguiente:

1. Identificar los vocablos que determinan las habilidades que ha de poseer cualquier candidato para la oferta en cuestión y extraer de Twitter aquellos tuits con contenido relacionado con ellos.
2. Dados esos tuits, construir un conjunto de usuarios, que entendemos como posibles candidatos a la oferta.
3. Usar la información publicada por los usuarios para determinar el grado de adecuación a la oferta (serán más adecuados aquellos que hayan publicado información sobre todos los conocimientos requeridos que aquellos que solo hayan publicado sobre alguno de ellos, y más relevantes aquellos más activos, según el número de tuits publicados sobre cada área).
4. Estudiar la relación entre los usuarios de este conjunto, y determinar los más relevantes en sentido relativo (en términos de actividad en la red, cuáles son los más “retuiteados”, cuáles los más seguidos, etc.).

1.2 Contexto de negocio

El área de Recursos Humanos de una compañía, incluso de una de las grandes¹, no es el primer área en la que suele pensarse cuando hablamos de aplicaciones de las técnicas Big Data en el mundo empresarial. En general, parece difícil que los departamentos de RRHH de las compañías, incluso de las mayores, produzcan tradicionalmente tanta información propia como para necesitar el software y las herramientas especiales que las técnicas Big Data proporcionan²: muchas compañías tienen miles, no millones, de empleados, y la información que recogen con respecto a ellos suele ser anual o cuatrimestral. En algunas ocasiones, el problema puede estar más incluso en poder acceder a los datos de forma unificada (por el uso de distintos sistemas de almacenamiento), y en entender y respetar las restricciones legales que aplican a los datos personales de los empleados a la hora de manejarlos y trasladar los resultados a otros departamentos.

¹O una empresa de *head hunting*, aunque en esta sección nos referimos solo a departamentos de RRHH.

²<https://hbr.org/2017/06/theres-no-such-thing-as-big-data-in-hr>

Sin embargo, es notorio el eco que este tipo de técnicas y su aplicación en el área de recursos humanos está teniendo recientemente, incluso en prensa generalista:

El País Transformación digital y Recursos Humanos: siete reflexiones

https://retina.elpais.com/retina/2017/12/19/tendencias/1513661663_430453.html

ABC El Big Data revoluciona los Recursos Humanos

http://www.abc.es/economia/abci-data-revoluciona-recursos-humanos-201609021552_noticia.html

Expansión Segundo estudio en España sobre transformación digital en RRHH

<http://www.expansion.com/blogs/lideres-digitales/2017/07/24/2-estudio-en-espana-sobre-transformacion.html>

ABC La revolución tecnológica llega a los recursos humanos pero necesita un cambio cultural

http://www.abc.es/tecnologia/informatica/soluciones/abci-revolucion-tecnologica-llega-recursos-humanos-pero-necesita-cambio-cultural-201706151516_noticia.html

Computerworld Meta4 reinventa su propuesta de gestión de recursos humanos

<http://www.computerworld.es/innovacion/meta4-reinventa-su-propuesta-de-gestion-de-recursos-humanos>

El Economista Por qué Recursos Humanos es un departamento clave en la transformación digital

<http://www.eleconomista.es/firmas/noticias/8523518/07/17/Por-que-Recursos-Humanos-es-un-departamento-clave-en-la-transformacion-digital.html>

Desde que los primeros departamentos de Recursos Humanos aparecieron (en Estados Unidos en los primeros años del siglo XX³), se ha estudiado mucho la labor y la función de estos departamentos dentro de las organizaciones empresariales, y las mejores técnicas y recursos para llevarlos a cabo, y existe abundante bibliografía al respecto (ver, por ejemplo <http://www.whatishumanresource.com/hrm-text-books> para una primera aproximación). Seguramente, la componente humana del profesional de RRHH no puede ser sustituida en numerosas circunstancias. Sin embargo, el cambio que las tecnologías digitales suponen en todos los ámbitos de la actividad humana sin duda impacta en las actividades de los departamentos de RRHH. El principal uso del Big Data y del análisis de datos debe ser la ayuda en la toma de decisiones, idealmente en todos los niveles organizativos. Y es

³<http://www.whatishumanresource.com/the-historical-background-of-human-resource-management>

en los departamentos de RRHH donde se toman muchas decisiones sobre el activo más valorable que compone las organizaciones: las personas. "People analytics" es el nombre que recibe la aplicación de las técnicas y tecnologías de Big Data y ciencia de datos al área de la gestión de personas⁴.

Es por ello que el proceso de transformación digital de las empresas también debiera incluir a los departamentos de Recursos Humanos, aprovechando los beneficios que les podría aportar un enfoque fundamentado en los datos para acometer su tarea. Puede encontrarse abundante discusión en la red acerca del papel que las tecnologías y técnicas Big Data pueden jugar en el área de RRHH⁵ y también recientes estudios que intentan formalizar y dar contexto a dicho papel [14]. Entre otras, algunos de los aspectos de la labor de los departamentos de RRHH en los que la aplicación de la tecnología Big Data puede ser de utilidad son los siguientes:

- Procesos de selección: incorporando datos de redes sociales y portales de empleo, los procesos de selección pueden ser más específicos y eficientes en encontrar candidatos adecuados a las posiciones ofertadas. Este punto tiene gran importancia, puesto que las pérdidas debidas a una mala contratación pueden ser cuantiosas⁶.
- Factores de calidad en la contratación: las técnicas Big Data pueden ayudar a los profesionales de RRHH a determinar qué factores en la trayectoria de un posible candidato son o no importantes a la hora de valorar su adecuación a un puesto dado y su rendimiento posterior. A veces, factores que en principio pueden percibirse como muy importantes (años de experiencia pasada, si esa experiencia fue en puestos similares o no,...) pueden en realidad ser irrelevantes a la luz de los datos disponibles en la compañía (y cada una, en cada sector y momento, es distinta).
- Formación y evaluación: no se trata solo de contratar a los mejores candidatos, sino de que las personas se adapten pronto a sus nuevas labores y al nuevo entorno de trabajo, y mantener una continua formación que permita al trabajador adaptarse a los cambios en dicho entorno. Con las técnicas de análisis de datos, puede ser más fácil decidir qué formación es la más adecuada para cada profesional, y evaluar los beneficios de dicha formación.

⁴También "workforce analytics", "talent analytics", "human capital analytics", "talent management analytics", entre otros.

⁵Entre muchos otros:

<https://www.business.com/articles/how-to-utilize-big-data-for-human-resources/>,
<https://blog.cake.hr/8-ways-use-hr-analytics-big-data-workplace/>,
<http://searchhrsoftware.techtarget.com/feature/Ready-or-not-here-comes-HR-analytics>,
<https://www.forbes.com/sites/forbeshumanresourcescouncil/2017/02/02/six-powerful-ways-your-hr-team-can-leverage-big-data/#282624d65de9>,
<http://www.digitalistmag.com/future-of-work/2017/08/31/5-powerful-ways-hr-can-leverage-big-data-05242673>,
https://www.villanovau.com/resources/hr/the-role-of-big-data-in-hr/#.WkVNst_ibtQ,
https://www.villanovau.com/resources/hr/big-data-changing-hr/#.Wkdf39_ibtQ,
<https://www.cornerstoneondemand.com/glossary/big-data-hr>

⁶<https://www.entrepreneur.com/article/244730>

- Retención de los empleados: evaluando el historial de la compañía, las técnicas de análisis de datos pueden ayudar a identificar las causas por las que un empleado deja de formar parte de la misma. A partir de ahí, se pueden identificar aquellos que no permanecerán en la compañía, así como aquellos factores que mejor funcionan para retener a los empleados, y potencialmente optimizar el uso de los recursos destinados a fidelización de empleados. Medir la percepción de la sociedad de la compañía como lugar de trabajo a través de las impresiones reflejadas en redes sociales, por ejemplo, también puede ayudar a los profesionales de RRHH a identificar áreas de mejora o áreas a potenciar, a la hora de atraer candidatos para nuevos puestos y retener a los trabajadores.
- Medida del desempeño: las técnicas de análisis de datos también pueden ayudar a medir el desempeño de los empleados de forma más precisa, por ejemplo determinando la relación entre franjas horarias trabajadas y rendimiento, identificando a los trabajadores más productivos de forma absolutamente objetiva, o llevando a cabo evaluaciones del desempeño más frecuentes que anualmente o trimestralmente. Otro aspecto importante en este punto es el uso de dispositivos encaminados a monitorizar la actuación de los trabajadores: por ejemplo, plataformas de escucha *online* para seguir en tiempo real el desempeño en un *call center*, o ayudar a los trabajadores en el tránscurso de su quehacer. Este tipo de dispositivos también puede ayudar a los departamentos de RRHH a cuidar estrechamente la seguridad del trabajador, el cumplimiento de las políticas de buenas prácticas, etc.
- Compensación monetaria: el dinero que ganan los empleados con su trabajo es un factor básico de la satisfacción del empleado. Usando el análisis de datos, los equipos de RRHH pueden desarrollar un modelo financiero optimizado. Esto es especialmente importante en compañías grandes, con empleados en distintas geografías, donde resulta difícil comparar los distintos departamentos internacionales. Y es importante en cualquier compañía tener una guía de cómo realizar una promoción adecuada de los trabajadores optimizando los costes asociados.
- Planificación: el análisis de datos puede ayudar a las compañías a identificar tendencias y planificar acciones. Por ejemplo, se pueden detectar épocas de mayor absentismo por cuestiones de salud e implementar políticas para paliarlo (de prevención sanitaria o de contratación de refuerzos), o detectar factores que conducen a una mayor incidencia de accidentes que pueden afectar a la producción y programar acciones para remediarlo.

En general, el uso de técnicas de análisis de datos y Big Data, ayudará a cambiar el enfoque de los departamentos de RRHH a un papel más proactivo, y a ampliar el rango de herramientas en las que apoyarse al tomar decisiones. En [28] las autoras reflejan varios ejemplos de compañías donde ya se está aplicando analítica avanzada para solucionar diversos problemas de los arriba mencionados.

El abanico de aplicaciones enfocadas a la gestión de Recursos humanos es bastante amplio (en <https://www.capterra.com/human-resource-software/> podemos encontrar una lista con varias aplicaciones y soluciones enfocadas a este tema), y van desde soluciones propuestas por

proveedores generales de tecnología Big Data y análisis de datos, como las de Oracle y SAP, a sistemas propuestos por proveedores más especializados, como SumTotal Systems, PeopleFluent, Cornerstone, Talent Analytics, CakeHR. Algunas compañías han desarrollado soluciones propias, por ejemplo el *Project Oxygen* de Google, enfocada a evaluar la actividad de los managers y ayudarlos en su desempeño.

Nuestro proyecto se encuadra dentro del primer punto mencionado en la página 12, y el objetivo es, como ya se ha mencionado, ayudar en la labor de localizar candidatos a un puesto de trabajo a través de la extracción y análisis de información de Twitter. Es por tanto un proyecto que clasificará perfiles de usuarios de Twitter.

Numerosos trabajos han explorado la clasificación de usuarios de Twitter atendiendo a su actividad (ver [4] y las referencias incluidas, [19]), para descubrir características de usuarios como etnia o género([15], <https://www.kaggle.com/crowdflower/twitter-user-gender-classification>), para clasificar la orientación política ([16]) o detectar usuarios similares adecuados para seguir ([17]), detección de bots ([18]), entre otros enfoques.

En cuanto a aplicaciones que extraigan información de Twitter y proporcionen algún tipo de clasificación de usuarios, podemos encontrar una amplia variedad. Algunas de las más comentadas en la red son las siguientes:

- StatusPeople.com: analiza los seguidores de una cuenta de Twitter y los clasifica como buenos, malos o inactivos.
- Mentionmapp: crea un mapa interactivo que dibuja por menciones, la interacción de una cuenta de Twitter con las últimas 200 publicaciones de la cuenta. Sirve para determinar la calidad de la audiencia en Twitter y poder detectar y diferenciar entre seguidores, fans y superfans.
- Audiense (antes SocialBro): es una herramienta para identificar audiencias relevantes (de dónde proceden los followers, en qué idioma escriben, si sus cuentas son públicas o privadas, si son cuentas verificadas, identifica seguidores inactivos, etc.). También examina el contenido psicolingüístico de los tuits y permite a las marcas conocer los rasgos de personalidad que están detrás del comportamiento de compra de los usuarios.
- Twitter Audit: analiza una cuenta tomando como referencia 5.000 seguidores del usuario y le asigna una puntuación según el número de tuits, la fecha del último tuit y la proporción de seguidores/seguidos. Y a continuación, asigna un porcentaje de "veracidad" de los seguidores de dicha cuenta.
- Untweeps: averigua qué seguidores de una cuenta no han tuiteado en los últimos 30 días, y facilita dejar de seguirles.
- Followerwonk: busca usuarios por palabras clave en sus biografías de los usuarios y los muestra por orden de influencia, pudiendo filtrar por localización, número de followers, número de Tweets, etc.

- Buzzsumo: esta herramienta sirve para localizar usuarios influyentes en Twitter a partir de palabras clave o para medir el rango de influencia de una persona o marca en concreto.
- Klear: también para localizar usuarios influyentes en diversas plataformas, además de Twitter.
- Trolldor: ofrece una lista negra de cuentas de Twitter clasificadas como falsos usuarios influyentes o trolls.
- Klout: puntuá el perfil social según su influencia y facilita una lista de los bios más influyentes según la temática. Propone contenidos para mejorar la puntuación de una cuenta.
- Peerindex: también ayuda a encontrar usuarios influyentes, en diversas redes sociales además de Twitter.

Hasta donde conocemos, no hay ninguna aplicación, comercial o no, ni trabajo académico, que tenga la misma orientación que el proyecto que desarrollamos. Hay, por supuesto, referencias en las que nos apoyaremos para clasificar el lenguaje de los tuits y descripciones, para determinar qué perfiles son personas, y para clasificar el texto de los tuits, así como para luego ordenar por importancia y relevancia los usuarios detectados (de todas estas referencias hablaremos más concretamente en cada punto particular). Sin embargo, la combinación de técnicas que proponemos en este proyecto no la hemos encontrado, y en particular, no la hemos encontrado enfocada al objetivo planteado: encontrar posibles candidatos relevantes para una oferta de trabajo a partir de la actividad relacionada en Twitter.

1.3 Objetivos y criterio de éxito

Como hemos mencionado anteriormente, nuestro objetivo final es proporcionar una lista de usuarios de Twitter que hayan publicado contenido relacionado con una oferta de trabajo, ordenados por relevancia, de forma que un reclutador pudiera usarla para contactar con ellos e incluirlos en su proceso de selección.

Nuestro criterio de éxito es por tanto conseguir dicha clasificación de los usuarios, y aportar valor con un método propio, particularizable y eficiente, para conseguirlo.

1.4 Hipótesis y limitaciones

La hipótesis fundamental que estamos haciendo al iniciar este proceso, es que la actividad en Twitter acerca de un determinado tema (por ejemplo, publicar algo relacionado con Python), supone que el usuario en cuestión tiene conocimientos sobre dicho tema (en nuestro caso, entenderíamos que ese usuario posee conocimientos de Python). Esto es cuestionable, por supuesto, pero también ponderable si tenemos en cuenta que la actividad no sea esporádica. Si un usuario publica sobre un tema en numerosas ocasiones, la hipótesis de que ese tema no le resulta ajeno, va cobrando fuerza.

Entre las limitaciones de las que adolece el proceso definido para llevar a cabo el proyecto, se encuentran las siguientes:

1. En general, no todos los posibles candidatos tienen por qué usar Twitter, y por tanto habrá muchos que queden directamente fuera de nuestro proceso.
2. Los tuits utilizados en el proceso están sujetos a una ventana temporal. Habrá muchos candidatos, usuarios de Twitter, que no aparezcan en nuestros registros, por no presentar actividad durante ese tiempo.
3. Twitter impone limitaciones en la cantidad de información a la que deja acceder, y por ello, también es posible que los usuarios pierdan visibilidad en este proceso, porque el contenido publicado por ellos no se encuentre entre el proporcionado por la red social durante el proceso de extracción de datos.

Otra limitación del proceso es que la información que obtenemos de la red es a nivel de usuario de Twitter. La dirección de correo o el nombre verdadero de la persona en cuestión, o cualquier dato que pudiera identificarla no está necesariamente disponible en la aplicación, salvo que el usuario lo haya querido hacer público explícitamente. Esta información, y la forma en que se utilice, es clave para la usabilidad del resultado del proyecto, en dos aspectos principales:

1. para que el reclutador pueda hacer uso de la información, la persona ha de estar identificada, lo suficientemente como para abrir un canal de comunicación entre el reclutador y el posible candidato.
2. Desde el punto de vista de la comercialización del resultado del proyecto, el hecho de identificar usuarios en una red social y usar esa información con fines lucrativos, ha de ser implementado de forma muy cuidadosa. El impacto de la Ley Orgánica de Protección de Datos de Carácter Personal (LOPD) es muy relevante en nuestro proyecto, y merece un apartado especial. Nos ocupamos de ello en la sección [1.4.1](#).

En relación al primer punto, evidentemente proporcionar un usuario de Twitter ya es abrir un canal de comunicación. Sin embargo, solo la información de las publicaciones del usuario no es suficiente para incluirlo en un proceso de selección, incluso antes del primer contacto entre reclutador y candidato, y el primero probablemente necesitará más información (por ejemplo un CV) para considerar al segundo. Una forma de solventarlo sería cruzar la información de Twitter (el nombre de usuario) con la contenida en otros portales (como LinkedIn, Facebook, Academia.edu, ResearchGate, Glassdoor, etc.), ya que a menudo el usuario de Twitter es parte de los datos consignados en los CV. Esta extensión del proyecto la hemos dejado deliberadamente fuera del planteamiento de este proyecto, aunque sería *conditio sine qua non* para una implementación comercializable del proyecto.

1.4.1 La Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal

La Agencia Española de Protección de Datos, AEDP, define un dato de carácter personal del siguiente modo:

Consultas más frecuentes (FAQS)

■ 3.3. ¿Qué es un dato de carácter personal? ¿Me podéis dar un ejemplo al respecto?

18-jul-2016

Dato de carácter personal es cualquier información concerniente a una persona física identificada o identificable. Se requiere la concurrencia de un doble elemento: la existencia de una información o dato, y que dicho dato pueda vincularse a una persona física identificada o identificable.

Así, tienen la consideración de dato de carácter personal, entre otros, el nombre, apellidos, documento nacional de identidad, información relativa a la salud, la dirección IP, la imagen, la huella dactilar o la información que se facilite a través de un GPS, cuando se refieran a una persona física identificada o identificable.

Ten en cuenta, que en la sociedad de la información actual, se manejan grandes cantidades de datos personales, y que en ocasiones, facilitamos muchos de ellos al navegar por Internet o con el uso de APPS.

Para más información puede consultar los informes jurídicos elaborados por esta Agencia:

Informe 0193/2008 sobre tratamiento de datos de GPS.

Informe 0334/2008 sobre tratamiento de datos de DNI o NIF.

Figura 1.4.1: AEDP preguntas frecuentes, <https://sedeagpd.gob.es/sede-electronica-web/vistas/infoSede/detallePreguntaFAQ.jsf;jsessionid=147951A8A98D206E87F2655B9E96E7EB?idPregunta=FAQ>

El perfil en Twitter de un usuario se puede considerar por tanto un dato de carácter personal, en tanto en cuanto permitiría localizar e identificar al usuario, tal vez a través de cruces de la información en Twitter con información adicional (por ejemplo, lo que comentábamos a propósito de encontrar el CV del usuario usando que frecuentemente, el usuario de Twitter es parte de la información contenida en el mismo, y el CV está disponible en otros portales). Sin embargo, en la política de privacidad de Twitter (<https://twitter.com/es/privacy>) se establece que

“Al utilizar cualquiera de nuestros Servicios, usted da su consentimiento para la recopilación, la transferencia, la manipulación, el almacenamiento, la revelación y otros usos de su información según lo descrito en esta Política de Privacidad. Esto incluye cualquier información que elija proporcionar que se considere sensible según la legislación vigente.”

Y también, en relación a la información del perfil o publicaciones:

“Información básica de la cuenta: si opta por crear una cuenta de Twitter, debe promocionar cierta información personal, como su nombre, nombre de usuario, contraseña, dirección de correo electrónico o número de teléfono. En Twitter, su nombre y nombre de usuario siempre se hacen públicos, incluso en su página de perfil y en los resultados de búsqueda, y puede utilizar su nombre real o un seudónimo.”

“Tuits, gente que sigue, listas, perfil y otra información pública: Twitter está principalmente diseñado para ayudarle a compartir información con el mundo. La mayoría de la información que usted nos facilita a través de Twitter es información que nos está pidiendo que hagamos pública. Puede facilitarnos información de perfil para hacerla pública en Twitter, como por ejemplo, una breve biografía, su ubicación, su sitio web, fecha de nacimiento, o una fotografía. Además, su información pública incluye los mensajes que tuitea; los metadatos facilitados con los tuits, tales como cuándo ha tuiteado y la aplicación cliente que utilizó para tuitear; información sobre su cuenta,

como el momento de su creación, el idioma, el país y la zona horaria; y las listas que crea, las personas a las que sigue, los tuits que retuitea o marca como Me gusta, y las emisiones de Periscope en las que hace clic o con las que se relaciona de alguna forma (por ejemplo, haciendo comentarios o clic en el icono de corazón) en Twitter. Twitter disemina amplia e instantáneamente su información pública a una amplia gama de usuarios, clientes y servicios, incluyendo motores de búsqueda, desarrolladores y editores que integran contenido de Twitter en sus servicios y organizaciones, tales como universidades, agencias de salud pública y empresas de investigación de mercado que analizan la información en busca de tendencias y conocimiento.”

A tenor de estas afirmaciones, la información que nosotros vamos a manejar en la implementación de este proyecto (nombre de usuario, información del perfil, tuits publicados) es una información de carácter público.

En su enunciado, la LOPD establece que (texto extraído del informe jurídico 2013-0184 de la AEPD http://www.agpd.es/portalwebAGPD/canaldocumentacion/informes_juridicos/otras_cuestiones/common/pdfs/2013-0184_Red-social-y-creaci-oo-n-de-perfiles-de-empleados..pdf)

Establece a este respecto la Ley Orgánica 15/1999 en su artículo 2 que "El régimen de protección de los datos de carácter personal que se establece en la presente Ley Orgánica no será de aplicación: a) A los ficheros mantenidos por personas físicas en el ejercicio de actividades exclusivamente personales o domésticas."

Y define las actividades personales a continuación:

En cuanto a la determinación de que se entiende por actividades personales o domésticas dispone el Reglamento de desarrollo de la LOPD en su artículo 4 que "Sólo se considerarán relacionados con actividades personales o domésticas los tratamientos relativos a las actividades que se inscriben en el marco de la vida privada o familiar de los particulares." Esta es también la interpretación del término "personal" contenida en la Sentencia de la Audiencia Nacional de 15 de junio de 2006 al señalar que "(...) Será personal cuando los datos tratados afecten a la esfera más íntima de la persona, a sus relaciones familiares y de amistad y que la finalidad del tratamiento no sea otra que surtir efectos en esos ámbitos.

También dará lugar a la aplicación de la Ley Orgánica 15/1999, por superar el ámbito de la vida privada o familiar de los particulares la publicación de datos de terceros en la red cuando no existan limitaciones de acceso a su perfil, en cuanto que dicha publicación constituye una cesión de datos, definida en el artículo 3 j) de la LOPD como "Toda revelación de datos realizada a una persona distinta del interesado", ya que en estos supuestos, como señalaba la sentencia la Sentencia de 6 de noviembre de 2003 (caso Bodil Lindqvist) del Tribunal de Justicia de las Comunidades Europeas, no se inscribe en el marco de la vida privada o familiar de los particulares "un tratamiento de datos personales consistente en la difusión de dichos datos por Internet de modo que resulten accesibles a un grupo indeterminado de personas."

Refiriéndose incluso a datos de carácter público que aparecen en los datos, la interpretación de la LOPD respecto al uso de los mismos por terceros, especialmente en el caso en el que la finalidad de dicho uso sea comercial, es que el único medio para legitimar el uso es el consentimiento explícito de la persona cuyos datos van a utilizarse. Este consentimiento ha de cumplir unas determinadas condiciones (de nuevo extraemos del informe jurídico 2013-0184 de la AEPD)

El tratamiento de datos de carácter personal debe encontrarse fundado en alguna de las causas legitimadoras previstas en el artículo 6 de la Ley Orgánica 15/1999, disponiendo a este respecto su número primero que "El tratamiento de los datos de carácter personal requerirá el consentimiento inequívoco del afectado, salvo que la ley disponga otra cosa."

(...)

Dicho consentimiento debe reunir las características señaladas en el artículo 3.h de la misma Ley que lo define como "manifestación de voluntad, libre, inequívoca, específica e informada, mediante la que el interesado consienta el tratamiento de datos personales que le conciernen".

Esta Agencia ha venido describiendo en sus informes dichas características de manera que se entiende por consentimiento libre aquel que ha sido obtenido sin la intervención de vicio alguno del consentimiento en los términos regulados por el Código Civil.

Selección de personal

El principio de finalidad y el consentimiento son elementos determinantes. El carácter público de una red social o del perfil de un usuario no legitima el acceso, la recopilación o el tratamiento de datos personales para cualquier tipo de finalidad.

No puede presumirse un consentimiento tácito por el hecho de la presencia en un entorno de red social profesional para cualquier tipo de tratamiento. No es lo mismo contactar y/o visualizar un perfil en una red profesional que incorporarlo a una base de datos de empleados potenciales.

Figura 1.4.2: Ayuda Ley de Protección de Datos, <https://ayudaleyproteccióndatos.es/2010/09/16/redes-sociales-empresas-y-protección-de-datos/>

Como consecuencia de toda esta información, entendemos lo siguiente:

- que los datos del perfil de los usuarios de Twitter, así como sus publicaciones en dicha red, tienen carácter público.
- Que el carácter público de dichos datos no es óbice para poder manipularlos y distribuirlos a terceros, en ninguna actividad que no se circunscriba al ámbito personal o familiar.
- La elaboración de un proyecto de fin de máster no tiene por qué entenderse como una actividad del ámbito personal o familiar, con lo cual no estaría en disposición de difundir esa información a terceros, salvo en las condiciones previstas en la LOPD.
- Cualquier versión comercializable de este proyecto debería contar con los mecanismos adecuados para obtener el consentimiento explícito de los usuarios para el uso de sus perfiles, y

posible inclusión en un proceso de selección de personal. Esta fase quedará fuera del plan de elaboración del proyecto.

Para eliminar el riesgo relativo a protección de datos en la elaboración del proyecto, pero sin alterar su esencia, proponemos publicar los resultados enmascarando los nombres de usuario de aquellos que aparezcan en nuestra base de datos. A pesar de no ser una solución perfecta (hay aplicaciones y páginas web que permiten obtener el identificador de usuario de Twitter o simplemente id⁷ a partir del nombre de usuario, y viceversa), hemos decidido que, salvo para la exploración inicial de los datos, usaremos siempre el id de Twitter de cada usuario para mostrar los resultados.

1.5 Esquema de desarrollo del proyecto

En las siguientes secciones de la memoria iremos estudiando y describiendo las distintas fases que componen el proyecto, que podemos dividir en tres grandes subgrupos:

1. Comenzaremos por hacer un análisis descriptivo de la base de datos recogida a partir del API de Twitter, en relación con el problema que nos ocupa. Estudiaremos la proporción de tuits originales frente a retuiteados, número de tuits descargados por día, número de retuits, relación entre el número de tuits descargados y el número de usuarios distintos, el número de tuits por usuario, y los hashtags presentes en los mensajes. Esta tarea la llevaremos a cabo en el capítulo 4.
2. A continuación, comienza la fase de proceso de la información. Nuestro objetivo es construir una lista de usuarios que podamos recomendar a un profesional de Recursos Humanos como posibles candidatos para una oferta. El primer paso, realizado en el capítulo 5, es entonces seleccionar, a partir de la información descargada de Twitter, los usuarios de interés.
3. La fase siguiente será, con esa lista de usuarios extraídos en la fase anterior, ordenarlos en función de su comportamiento en Twitter (difusión e interacción con otros usuarios). La metodología empleada en este apartado, así como una descripción de los resultados, se abordan en el capítulo 6.
4. Por último, pero no menos importante, desarrollaremos la fase de visualización de los datos, para aportar usabilidad a los resultados del proyecto, fase descrita en el capítulo 7

Este esquema corresponde con el esquema de la llamada general a las distintas funciones que se ejecutan en el proyecto (en el fichero **master.py** del repositorio, ver la sección 3.1):

⁷Con "id" nos referiremos al identificador único que asigna Twitter a cada usuario, que está asociado a la cuenta del usuario, <https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/user-object>, y no puede ser modificado por él salvo creando una cuenta distinta.

```
def run_all_processes():
    data_base_name = "query1_spanish_stream"
    set_up_path = "keys/set_up.py"

    # descriptives
    run_descriptives(data_base_name, set_up_path)

    # db_to_table
    all_tweets_data = create_tables(data_base_name, set_up_path)
    # user_selection
    selected_users = select_users(all_tweets_data)
    file_path = "tables/3_selected_users.xlsx"
    save_df(selected_users, file_path = file_path)
    # user_rank
    ranked_users = get_ranked_users(selected_users)
    file_path = "tables/4_ranked_users.xlsx"
    save_df(ranked_users, file_path = file_path)

    print ("Finished Process, data ready for visualization")
```

Figura 1.5.1: Llamadas a las funciones de procesado de la información

Capítulo 2

Planificación del proyecto

2.1 Equipo

El equipo de Octopus Data Insights está formado por cuatro personas, con perfiles multidisciplinares y complementarios:

- Fabio Inui: informático, con cinco años de experiencia en programación de sistemas como consultor y quince años de experiencia en el sector de banca con tarjetas de pago y financiación, de ellos cuatro años en MIS con construcción de DataMarts y reportes, siete años dedicados a business intelligence con SAS y Oracle PL/SQL para áreas de CRM, Productos, Financiera y Crédito, y cuatro años en planificación financiera.
- Teresa Martínez: matemática, con diez años de experiencia en investigación y docencia a nivel universitario, ocho en construcción de modelos de valoración de derivados en empresa financiera de primer nivel, y cuatro de gestión de fondos en una de las principales gestoras españolas.
- Javier Quintana: más de 17 años de experiencia en el sector de las Telecomunicaciones y responsabilidades en Desarrollo Comercial, Marketing de Producto y Canales, Inteligencia Competitiva y Consultoría Estratégica, entre otras, y con foco en el segmento de Empresas, principalmente.
- Silvia Santos: ingeniero industrial, con más de 15 años de experiencia en entornos productivos del sector industrial y 3 como consultora de Facility Management.

2.2 Desarrollo temporal

El desarrollo del proyecto se ha llevado a cabo a lo largo de varias fases:

- **Obtención de datos:** esta primera fase se desarrolló a finales de Noviembre de 2017, y se dividió en dos subtareas:

- La recogida de tuits se llevó a cabo a lo largo del periodo entre el 18 de Noviembre y el 1 de Diciembre de 2017.
 - En los días posteriores se llevó a cabo el análisis inicial de los tuits.
- La fase de **limpieza de los tuits** se llevó a cabo entre el 8 de diciembre de 2017 y el 29 de Enero de 2018, y se dividió en tres subtareas que llevamos a cabo en paralelo:
- Algoritmo para detección idioma
 - Algoritmo para la detección del tema de referencia
 - Detección de usuarios relevantes (personas frente a empresas, bots)
- Construcción del **modelo de ordenación de candidatos**: la fase final del proyecto se ha desarrollado a lo largo del mes de Febrero.
- Por último, se lleva a cabo la **generación de informes y presentación de resultados** de cara a la entrega del proyecto.

La redacción de la memoria es una tarea continua, incorporando los resultados y desarrollos a medida que se han ido produciendo.

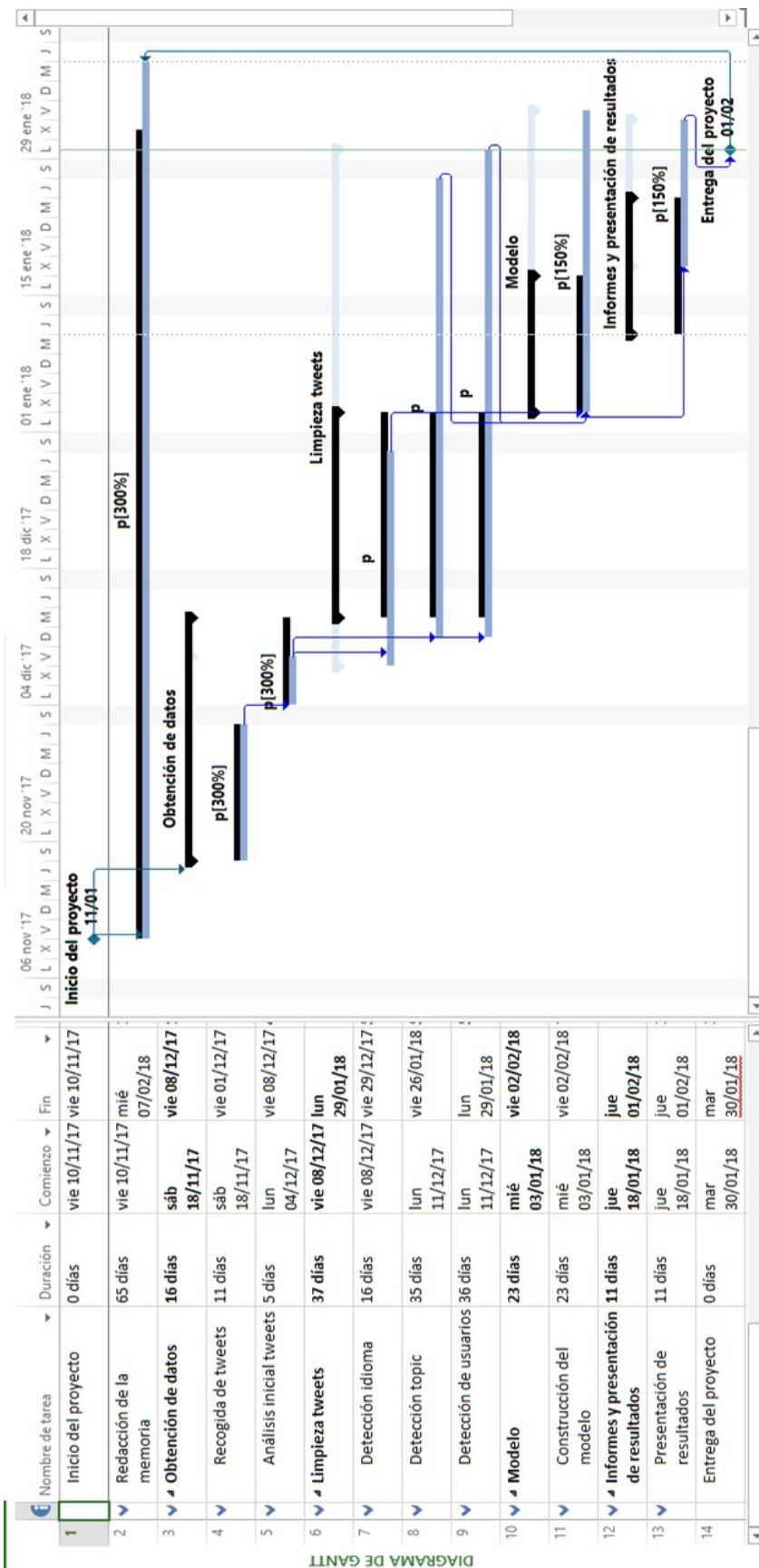


Figura 2.2.1: Planificación del proyecto.

Capítulo 3

Infraestructura

En esta sección describiremos la infraestructura que hemos construido para el desarrollo del proyecto.

3.1 Repositorio GIT

El código del proyecto, así como las presentaciones y memoria de este proyecto, está almacenado en el repositorio https://github.com/MaiteMartinez/MBITProject_Data4all

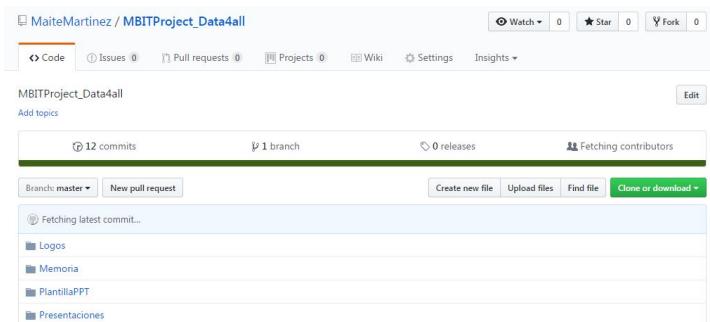


Figura 3.1.1: Repositorio del código del proyecto.

3.2 Infraestructura para la obtención de datos

Como muchas redes sociales, Twitter facilita el acceso a la información que generan sus usuarios a través de un API (*Application Programming Interface*)^[7]. El API de Twitter ofrece diversas opciones: Webhook APIs, ADS API, REST APIs y Streaming APIs. La primera está enfocada a generar notificaciones instantáneas a partir de detección de eventos y la segunda a la integración de aplicaciones con la plataforma de publicidad de Twitter. Para nuestro proyecto, solo son relevantes por tanto las dos segundas:

- El API Rest (*Representational State Transfer*) permite realizar consultas puntuales con los parámetros de búsqueda indicados, a través de una componente denominada Search API. El Search API funciona de manera similar, aunque no exactamente igual, a la búsqueda en

la página web de Twitter. El Search API realiza la búsqueda entre una muestra de tuits publicados en los últimos siete días, y las búsquedas están limitadas a 180 peticiones cada ventana temporal de 15 minutos.

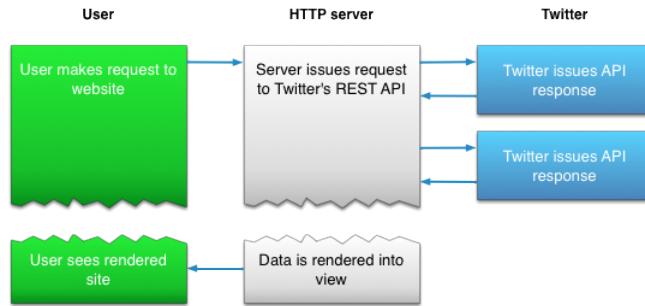


Figura 3.2.1: Funcionamiento del API Rest. <https://dev.twitter.com/streaming/overview>

La búsqueda realizada por este API está centrada en la relevancia y no en la completitud, lo que quiere decir que algunos tuits y usuarios podrían quedarse fuera.

- El API Streaming permite un acceso con baja latencia al flujo global de tuits de la aplicación, y requiere una conexión HTTP continua. Entre los tipos de flujos disponibles, en la web de Twitter para desarrolladores, se recomienda usar los flujos públicos para realizar minería de datos (en dichos flujos aparecen muestras de los datos públicos de Twitter).

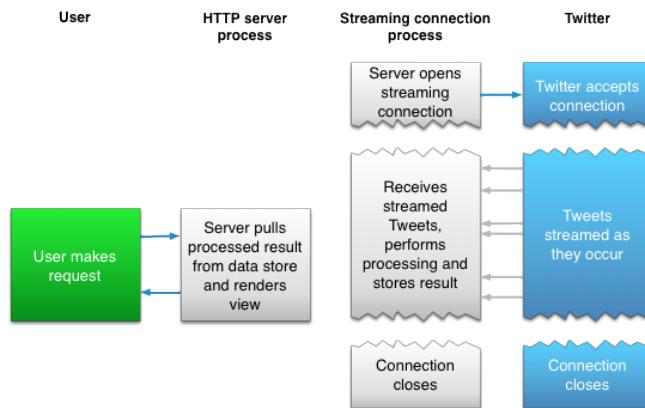


Figura 3.2.2: Funcionamiento del API Streaming. <https://dev.twitter.com/streaming/overview>

El acceso a ambas versiones de API está gobernado por la autentificación mediante el protocolo OAuth (*Open Authorization*), lo que implica que para cada aplicación deben obtenerse los tokens necesarios de la sección de desarrolladores de Twitter, estableciéndose un número máximo de 7 por usuario. También para ambas versiones del API existen restricciones de acceso. Estas restricciones solo afectan a las versiones gratuitas de los APIs. Hay una versión de pago de este acceso (Twitter

Firehose) que garantiza como respuesta el 100% de los tuits que cumplan los criterios de la búsqueda. Para el desarrollo de este proyecto nos hemos servido del API gratuito, por limitación de costes.

Entre los API REST y Streaming, hemos decidido utilizar el API Streaming, mediante un script en Python usando Tweepy. Este método tiene diversas ventajas y desventajas que pasamos a revisar:

1. El API Streaming proporciona tuits en tiempo real, y funciona como una especie de “grabadora”, con la que se van registrando todos los tuits a medida que se van produciendo.
2. Como es un proceso que se mantiene a la escucha y va reflejando entradas según se producen, la infraestructura necesaria para que funcione es algo más complicada que para el API Search. Se necesita, por ejemplo, una conexión continua a Internet, ya que el tiempo que el proceso no esté corriendo, no estaremos recogiendo tuits.
3. El límite de bajada de los tuits en el API Stream es de 50 tuits por segundo. En nuestro caso, el número de tuits que se producen no pasan de unas decenas por minuto, con lo cual nos aseguramos un barrido bastante exhaustivo de la actividad relevante.
4. Además, al ir bajando tuits consecutivamente, minimizamos el problema de tener tuits repetidos.
5. Como desventaja, no podemos acceder a tuits antiguos, y solo tendremos aquellos que se produzcan en el tiempo que esté el proceso de “escucha” levantado.

Con respecto al segundo punto, el script **Python/download_tweets_stream.py** de nuestro repositorio de GitHub es el encargado de bajar los tuits. Este script de Python usa la librería Tweepy, y va almacenando los tuits en una base de datos MongoDB a medida que el proceso de escucha los va captando.

```
#Listener
class StreamListener_to_db(StreamListener):
    def __init__(self, collection):
        self.collection = collection
        print("***** New stream created at " + str(datetime.datetime.now()))

    def on_data(self, data):
        tweet_json = json.loads(data)
        try:
            tweetid = self.collection.insert_one(tweet_json).inserted_id
            print ("===== twit loaded " + str(tweetid) +" at "+str(datetime.datetime.now()))
            return True # keep stream alive
        except BaseException as e:
            print ("failed ondata, " + str(e))
            time.sleep(5) # reload stream

    def on_error(self, status):
        text_error = "***** ERROR ** Status code = %s at %s\n" % (status_code,datetime.datetime.now())
        print (text_error)
        return True # keep stream alive

    def on_exception(self,exception):
        text_error = "***** EXCEPTION ** %s at %s\n" % (exception,datetime.datetime.now())
        print (text_error)
        return True # keep stream alive

    def on_timeout(self):
        #print "pass on_timeout"
        text_error = "***** TIMEOUT at %s\n" % ( datetime.datetime.now())
        print (text_error)
        return False #restart streaming
```

Figura 3.2.3: Función de escucha de tuits.

Para mantener vivo el proceso de escucha y que no caiga frente a errores o timeouts, incluimos en el script un control de incidencias:

```

exit = False
while not exit: # Making permanent streaming with exception handling
    try:
        stream = Stream(auth, 1)
        stream.filter(track=query, languages = ["es"])
    except KeyboardInterrupt:
        print ('\nGoodbye! ')
        exit = True
    except:
        print ("Error. Restarting Stream.... ")
        time.sleep(5)

```

Figura 3.2.4: Gestión de incidencias en el proceso de escucha.

Y para refrescar el proceso, y que la conexión con Twitter funcione correctamente de forma continua, se programaron relanzamientos diarios del script en el programador de tareas de Windows (a través del ejecutable **Python/streaming_process_keeper/streaming_upload.bat**, también en nuestro repositorio de GitHub).

3.3 Lenguajes de implementación

Para el desarrollo de la mayor parte de la lógica del proyecto hemos elegido Python 3.6 (a través de la instalación de Anaconda). Esta elección tiene diversas ventajas, en las varias fases del proyecto, ya que este lenguaje facilita las siguientes tareas:

- Comunicación con el API Streaming de Twitter a través del paquete Tweepy.
- Sencilla interacción con el formato JSON, que es en el que los tuits son descargados desde el API, gracias al paquete json.
- Comunicación con la base de datos documental MongoDB, a través del paquete pymongo.
- Incorporación nativa del encoding UTF-8 (“Unicode Transformation Format” con números de 8 bits), lo que nos ahorra muchos quebraderos de cabeza al tratar con caracteres especiales del español (tildes, ñ, etc.) y la presencia de emojis en los textos de los tuits.¹

En la parte de visualización hemos recurrido al paquete Shiny de R.

3.4 Almacenamiento y manipulación de los datos

Para la primera toma de contacto con los datos descargados del API de Twitter, nos hemos decantado por almacenarlos según íbamos recogiéndolos en una base de datos MongoDB. Las razones son varias:

- MongoDB trata de forma natural documentos en formato JSON.
- También maneja sin problemas documentos con encoding UTF-8², lo que nos ayuda a no tener problemas de encoding al almacenar los tuits.

¹<https://docs.python.org/3/howto/unicode.html>.

² <https://docs.mongodb.com/v3.4/reference/bson-types/>

- Se integra muy bien en programas en Python gracias al paquete pymongo.
- No necesita una definición de la estructura del documento, lo que es muy conveniente para tratar tuits, donde el número de campos que obtenemos del API y el contenido de esos campos no siempre sigue el mismo formato. Este punto es en particular relevante si el API de Twitter cambiase, o se introdujeran nuevos campos. Por ejemplo: a partir del 26 de Septiembre de 2017, Twitter cambió el límite de 140 caracteres a un límite de 280 caracteres a algunos usuarios, y eso provocó que aparecieran nuevos campos en el cuerpo de cada tuit³.
- Es una base de datos bastante rápida, y escalable.

La tabla final con los usuarios ordenados, así como diversos resultados intermedios, están almacenados en tablas de Excel. Lo hemos hecho así porque la exportación a cvs, por ejemplo, nos daba problemas con el encoding UTF-8 de varios campos como los textos de los tuits y los nombres de los usuarios.

3.5 Fuentes de información

Gran parte de este trabajo se ha basado en la investigación llevada a cabo por los miembros del equipo en la red, y en el manejo de las referencias encontradas en ella (la mayoría incluidas en la bibliografía). No podríamos haber llevado a cabo el proyecto sin la ayuda de herramientas como Google, YouTube y Stack Overflow.

³<https://developer.twitter.com/en/docs/tweets/tweet-updates>

Capítulo 4

Obtención y tratamiento inicial de los datos: análisis descriptivo

4.1 Descripción de los datos

En pantalla, un tuit relevante para nuestro proyecto podría ser el mostrado en la figura adyacente. Sin embargo, cuando descargamos el mismo tuit a través del API Search de Twitter, obtenemos mucha más información, en formato JSON.

El formato JSON (*Javascript Object Notation*)¹ es un formato ligero de intercambio de datos, fácilmente interpretable (por humanos y por máquinas). Es un formato ampliamente utilizado, siendo numerosos los lenguajes que son capaces de usar este formato (lenguajes de la familia C, Javascript, PHP, Python, etc.). En JSON se pueden representar dos tipos de estructuras: un conjunto de pares (clave,valor) con la sintaxis `{clave1:valor1, clave2:valor2,...}`, también denominado *objeto*, y un conjunto ordenado de valores con la sintaxis `[valor1, valor 2,...]` (que se denomina *arreglo*). Un valor puede ser una cadena de caracteres con comillas dobles, un número, un valor booleano o nulo, un objeto o un arreglo. Esta flexibilidad permite representar datos de gran complejidad.

Como veremos en el siguiente ejemplo de un tuit en formato JSON descargado a través del API Search de Twitter, puede resultar de ayuda pensar en un JSON como en un diccionario `{clave:valor}`,



Figura 4.1.1: Ejemplo de tuit.

¹<http://www.json.org/json-es.html>

donde las claves son cadenas de texto, y el valor es algo flexible que acomoda desde una cadena de texto a un vector de objetos o un nuevo diccionario. En particular, la información del tuit que considerábamos más arriba luce de la siguiente manera:

```
{'_id': ObjectId('59e0cbb03842ed08188233d7'),  
 'contributors': None,  
 'coordinates': None,  
 'created_at': 'Wed Oct 11 08:13:41 +0000 2017',  
 'entities': {'hashtags': [{'indices': [90, 106], 'text': 'MachineLearning'},  
                          {'indices': [107, 114], 'text': 'Python'}],  
             'symbols': []},  
 'urls': [{'display_url': 'twitter.com/i/web/status/9...',  
           'expanded_url': 'https://twitter.com/i/web/status/918026805080182785',  
           'indices': [116, 139],  
           'url': 'https://t.co/1WuwNRzn8z'}],  
 'user_mentions': []},  
 'favorite_count': 1,  
 'favorited': False,  
 'geo': None,  
 'id': 918026805080182785,  
 'id_str': '918026805080182785',  
 'in_reply_to_screen_name': None,  
 'in_reply_to_status_id': None,  
 'in_reply_to_status_id_str': None,  
 'in_reply_to_user_id': None,  
 'in_reply_to_user_id_str': None,  
 'is_quote_status': False,  
 'lang': 'es',  
 'metadata': {'iso_language_code': 'es',  
             'result_type': 'recent'},  
 'place': None,  
 'possibly_sensitive': False,  
 'retweet_count': 0,  
 'retweeted': False,  
 'source': '<a href="http://twitter.com" rel="nofollow">Twitter Web Client</a>',  
 'text': 'Vamos preparando el siguiente libro a estudiar que al final la movida  
 me está gustando... #MachineLearning #Python... https://t.co/1WuwNRzn8z',  
 'truncated': True,  
 'user': {'contributors_enabled': False,  
          'created_at': 'Sat Dec 12 09:13:46 +0000 2009',
```

```
'default_profile': False,
'default_profile_image': False,
'description': 'Me gustan las camisetas y las zapatillas // Desarrollo y '
    'Diseño para sistemas Apple // Creador de @GetPomodoroApp · '
    '@GetAtentoApp · @MADatBUS · @GetMeteo y...',
'entities': {'description': {'urls': []},
    'url': {'urls': [{'display_url': 'desappstre.com',
        'expanded_url': 'http://desappstre.com',
        'indices': [0,23],
        'url': 'https://t.co/oYY42NIHrT'}]}},
'favourites_count': 1512,
'follow_request_sent': False,
'followers_count': 211,
'following': False,
'friends_count': 209,
'geo_enabled': True,
'has_extended_profile': True,
'id': 96309647,
'id_str': '96309647',
'is_translation_enabled': False,
'is_translator': False,
'lang': 'es',
'listed_count': 109,
'location': 'Madrid — Mundo Real™',
'name': 'Adolfo ™',
'notifications': False,
'profile_background_color': '000000',
'profile_background_image_url': 'http://abs.twimg.com/images/themes/theme2/bg.gif',
'profile_background_image_url_https': 'https://abs.twimg.com/images/themes/theme2/bg.gif',
'profile_background_tile': False,
'profile_banner_url': 'https://pbs.twimg.com/profile_banners/96309647/1501577205',
'profile_image_url': 'http://pbs.twimg.com/profile_images/888396793024794624/O6gHh-
IJ_normal.jpg',
'profile_image_url_https': 'https://pbs.twimg.com/profile_images/888396793024794624/O6gHh-
IJ_normal.jpg',
'profile_link_color': '1B95E0',
'profile_sidebar_border_color': '000000',
'profile_sidebar_fill_color': '000000',
'profile_text_color': '000000',
'profile_use_background_image': False,
```

```
'protected': False,  
'screen_name': 'FitoMAD',  
'statuses_count': 6893,  
'time_zone': 'Madrid',  
'translator_type': 'none',  
'url': 'https://t.co/oYY42NIHrT',  
'utc_offset': 7200,  
'verified': False}}}
```

La descripción de cada campo de los que integran el tuit puede encontrarse en la página web de Twitter para desarrolladores, <https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/tweet-object>.

4.2 Obtención de los datos

Los tuits que componen nuestro corpus de datos los hemos obtenido a través del API Streaming de Twitter, a través de una búsqueda dirigida en el API Streaming. Esta búsqueda dirigida se ha realizado a través de palabras clave, asociadas a la actividad de data science, concretamente:

```
"machine learning" "machinelearning" "datamining" "data mining" "Python"  
"SQL" "hadoop" "bigdata" "big data" "pentaho"  
"rstats" "SAS" "tableau"
```

Esta petición se define como un vector en Python, donde la coma significa “OR” y el espacio dentro de las comillas significa “AND”, y se incluyeron dichos términos con y sin almohadilla (#). Las palabras clave de la búsqueda se almacenan en un fichero de texto **query.txt**, de tal forma que la extensión del proyecto a búsquedas relacionadas con otras ofertas de trabajo pueda realizarse de una forma sencilla y directa, sin cambiar el código.

También hemos incluido en la búsqueda un filtro por idioma, incluyendo el parámetro “languages = [“es”]” en la llamada al API, con el objetivo de bajar solo tuits en un idioma. En principio² esta búsqueda debería devolver tuits que la aplicación Twitter ha detectado como escritos en idioma español. Sin embargo, también bajamos tuits en otros idiomas (inglés, sobre todo), lo que nos obligará a incluir esta variable en el proceso de selección de usuarios, como veremos en la sección 5.1.

El script en el que se realiza la llamada al API de Twitter y el primer almacenamiento de los tuits es el script llamado **download_tweets_stream.py**. Este script importa otros de nuestro proyecto, como **OpenMongoDB.py**, que gestiona la conexión a la base de datos MongoDB en el que se almacenarán los tuits. El lanzamiento programado de la tarea se hizo con una entrada en el gestor de Tareas Programadas del portátil, a través del archivo **streaming_upload.bat**. Todos estos archivos se encuentran en el repositorio de GitHub descrito en la sección 3.1.

²<https://developer.twitter.com/en/docs/tweets/filter-realtime/guides/basic-stream-parameters>

4.3 Almacenamiento

Según van produciéndose los tuits, y nuestra “grabadora” los va detectando, los hemos almacenado en una base de datos de MongoDB, en local.

4.4 Revisión inicial de los datos

Una vez almacenados los tuits, realizamos un análisis exploratorio para estudiar con qué material contábamos para el desarrollo del proyecto. Este estudio se lleva a cabo en el archivo **descriptives.py** del repositorio de GitHub.

Tuits repetidos

Aunque hemos usado una conexión con el API Streaming de Twitter, que va almacenando los tuits según van publicándose, de los 14.736 documentos que hemos recogido en la base de datos, solo 14.727 son únicos, y hay 9 repetidos. Esto es extraño, pero podría deberse a dos procesos de Streaming corriendo en paralelo sobre la misma base de datos momentáneamente. Como son muy pocos, no los hemos eliminado para hacer este estudio previo de los datos.

El texto de los tuits

Nuestra primera duda es si los tuits que hemos bajado corresponden realmente a contenido relacionado con la ciencia de datos. Esto será objeto de una sección propia a la hora de limpiar los datos, pero de momento, una forma gráfica de ver si los textos de los tuits se corresponden con el tema que nos interesa es construir una nube de palabras con ellos. Aparentemente no hay grandes desviaciones del tema a tratar, ciencia de datos y big data:



Figura 4.4.1: Texto de los tuits

Tuits originales frente a tuits retuiteados

Nos interesaba saber qué proporción de tuits de los que hemos bajado son tuits originales y qué proporción son tuits retuiteados. En general, en los documentos obtenidos a través del API Streaming, Twitter marca con un “RT” al comienzo del texto del tuit aquellos que son retuiteados, a la vez que incluye en el cuerpo del tuit la información completa sobre el tuit que ha sido retuiteado. Hemos observado que hay tuits que parece que son retuits de otros (que comienzan por “RT”), pero luego no tienen los campos “retweeted_status” o “quoted_status”. Estos podrían ser mensajes procedentes de bots, que los envían de forma automática. De los 14.736 tuits, hay 6.421 originales, 8.048 retweets estándar y 267 aparentes retuits que no tienen información de lo retuiteado.

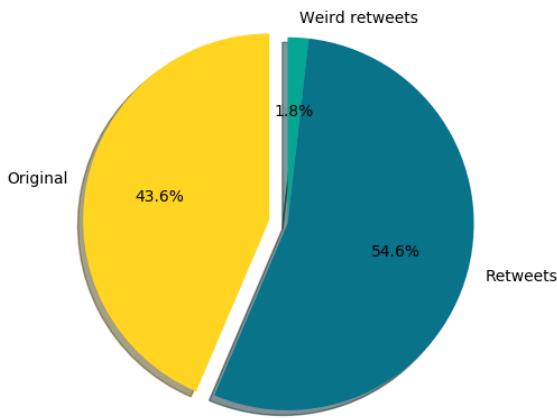


Figura 4.4.2: Proporción de tuits originales y retuiteados.

Número de retuits por tuit

A la vista del gráfico anterior, parece que la actividad principal que hemos captado es una actividad de difusión, en la que los usuarios retuitean información de otros usuarios. Entre estos tuits retuiteados, nos interesa estudiar el número de veces que cada tuit ha sido retuiteado. Nos vamos a fijar en aquellos tuits que aparecen como retuiteados en nuestra muestra, es decir, aquellos tuits cuya información viene en los campos “retweeted_status” o “quoted_status”, ya que son indicativos de los temas que interesan a los usuarios. En los 8.048 tuits de nuestro corpus que son retuits con info de lo retuiteado, se han retuiteado 2.686 tuits distintos. La distribución del número de retuits por cada uno de estos 2.686 aparece en las siguientes gráficas.

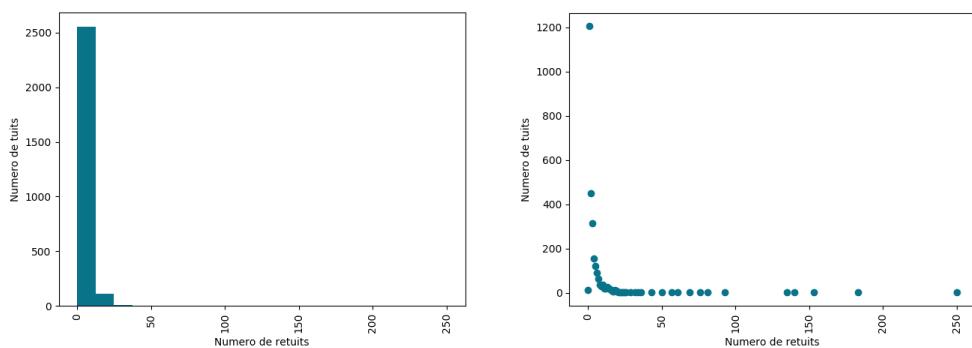


Figura 4.4.3: Retuits: número de retuits por tuit.

Tanto en el histograma con en el gráfico de puntos, es claro que la mayoría de tuits han sido retuiteados unas pocas veces (entre 1 y 5), y que solo unos pocos han sido retuiteados muchas veces. Los dos más tuiteados aparecen en la siguiente figura:



Figura 4.4.4: Retuits: los dos tuits más retuiteados.

Algo importante es que a partir de este análisis, parece que podemos encontrar usuarios relevantes para nuestro proyecto no solo en los usuarios que han publicado los tuits que hemos descargado, sino también en los usuarios que han publicado los tuits que los primeros han retuiteado.

Tuits descargados a lo largo del tiempo

Hemos descargado tuits durante dos semanas a finales de Noviembre de 2017. La distribución temporal del número de tuits obtenidos es la siguiente:

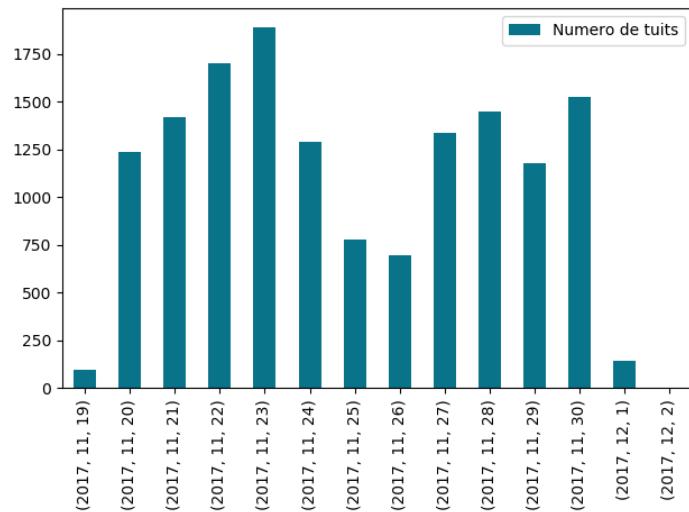


Figura 4.4.5: Tuits descargados diariamente.

El día 19 de Noviembre fue domingo, parece por tanto que se aprecia cierto patrón estacional en la actividad (tuiteándose más entre semana que en fin de semana). Los días 22 y 23 de Noviembre hubo un evento BigData, el V Encuentro de Big Data en Castilla y León, que tal vez tenga relación con la mayor actividad durante esos días.

Relación entre número de tuits descargados y número de usuarios distintos

Para nuestro proyecto, lo más relevante de la base de datos con la que trabajamos es qué información sobre los usuarios podemos extraer a partir de los tuits almacenados. El primer paso suele ser contar, y por ello miramos cuántos usuarios distintos podemos obtener de ella. El siguiente gráfico representa el número de usuarios distintos en función del número de tuits descargados, a lo largo del tiempo.

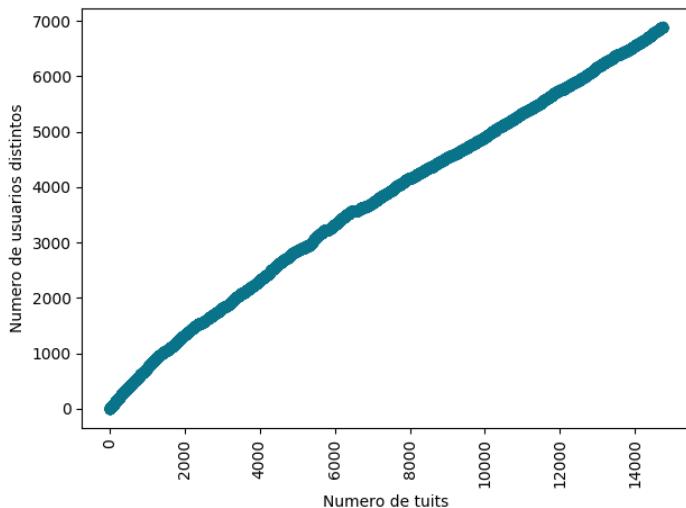


Figura 4.4.6: Número de usuarios distintos frente a número de tuits descargados diariamente.

Este gráfico no presenta sorpresas, y se aprecia que la relación entre el número de tuits descargados y el número de usuarios distintos es prácticamente lineal.

Tenemos 6.890 usuarios distintos (en los usuarios de los tuits descargados, sin contar los usuarios de los tuits que estos usuarios hayan podido retuitear), lo que arroja una media de unos 2.13 tuits por usuario. Ya sabemos que la media es una medida muy poco robusta. Si queremos hacernos una idea de la actividad de nuestros usuarios, mejor estudiamos un poco más la distribución de esa variable.

Número de tuits por usuario

Al igual que hicimos con el número de retuits por tuit retuiteado, veamos un poco más en detalle la distribución del número de tuits que cada usuario ha publicado:

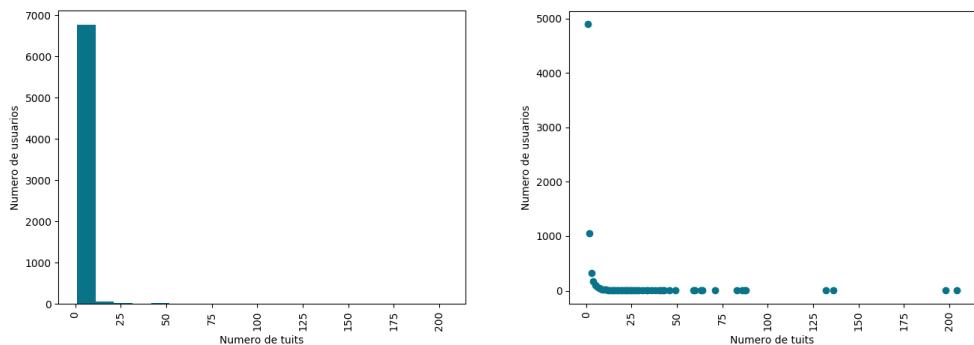


Figura 4.4.7: Número de tuits por usuario.

También en este caso se aprecia que la mayoría de los usuarios han tuiteado una vez, pero que hay algunos usuarios con un número muy elevado de tuits:

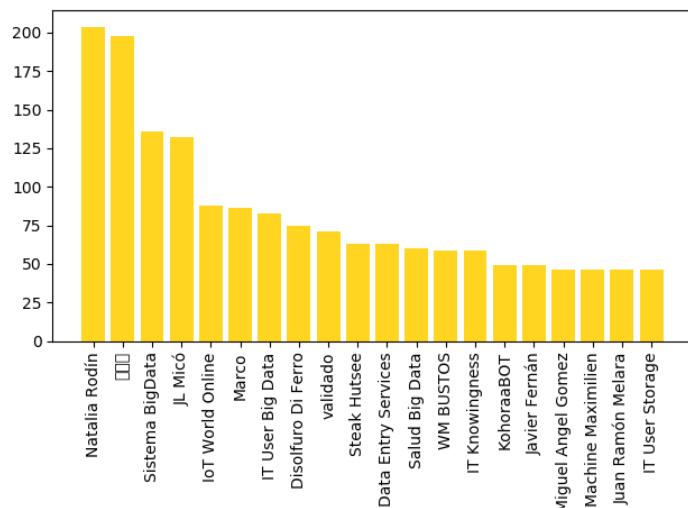


Figura 4.4.8: Número de tuits de los usuarios más activos.

De estos usuarios:

- Natalia Rodín, Sistema BigData, Steak Hutsee parece que ya no existen en Twitter
- IoT World Online es un grupo de personas interesadas en el mundo del Big Data.
- IT User Big Data, validado, Data Entry Services, IT User Storage no está claro si son personas (solo retuits, sin bio o con bio poco descriptiva).
- Salud Big Data es un portal de noticias
- IT Knowingness es un feed
- KohoraaBot, Machine Maximilien son bots

En resumen, solo ocho de los veinte usuarios con mayor actividad parecen adecuados para nuestro proyecto. Esto es indicativo de que la labor de filtrado de la base de datos va a ser clave para conseguir un producto exitoso.

Localización de los tuits

Desde el punto de vista de la comercialización del producto, parece relevante incorporar información geográfica acerca de los candidatos. Para ello, siguiendo el enfoque de este trabajo, deberíamos usar información de este tipo extraída del corpus de tuits descargados de Twitter. Esto nos lleva a preguntarnos cuántos de ellos tienen el dato disponible.

Según la documentación del API de Twitter³, los tuits pueden asociarse con una localización, generando un tuit que ha sido “geolocalizado”, y estas localizaciones pueden ser un punto exacto (con coordenadas de longitud y latitud) o un lugar como una ciudad o país o un área definida mediante una “bounding box”. Hay dos campos en un tuit que se usan para describir la geolocalización: “coordinates” y “place”. El objeto “place” siempre está presente en el caso de que el tuit esté geolocalizado, mientras que las coordenadas solo en el caso en el que el tuit tenga asignada una localización exacta. En el siguiente gráfico hemos representado el porcentaje de tuits con el campo “place” con valores, en el que podemos apreciar que, salvo que cambiásemos la búsqueda en Twitter, no es un campo que vayamos a poder usar en el análisis:

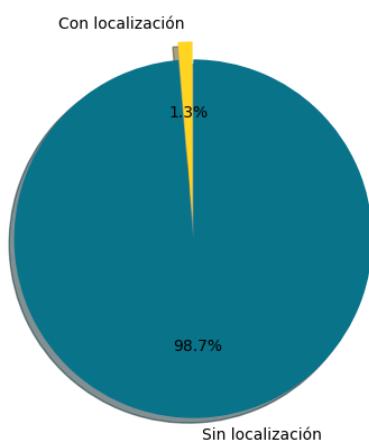


Figura 4.4.9: Proporción de tuits geolocalizados.

Hashtags: distribución

En nuestro proyecto, va a ser muy importante cómo clasifiquemos los contenidos de los tuits. Los propios usuarios clasifican los contenidos de los tuits cuando los etiquetan a través de los denominados “hashtags”, que son palabras precedidas por el símbolo # (almohadilla o hash).

Hemos estudiado los hashtags incluidos en los tuits de la base de datos, tanto los de los tuits descargados, como de aquellos citados o retuiteados en ellos. Los siguientes gráficos dan una idea de los hashtags presentes en ambos casos:

³<https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/geo-objects>



Figura 4.4.10: Hashtags en los tuits descargados (arriba) y en los tuits descargados y citados y retuiteados en ellos.

En principio parece que no difieren mucho. Vamos a comparar los hashtags más frecuentes en ambos conjuntos de tuits:

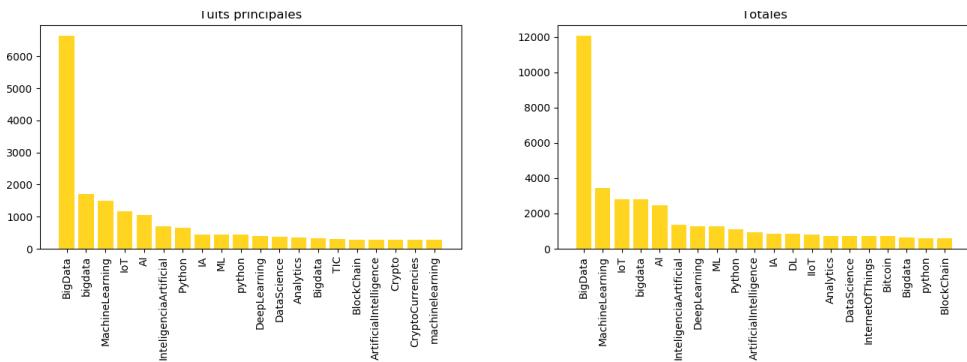


Figura 4.4.11: Hashtags más frecuentes en los tuits descargados (izq.) y en los tuits descargados y citados y retuiteados en ellos.

Perfiles de los usuarios

A la hora de desarrollar el proyecto y construir la lista de usuarios recomendados, es importante filtrar aquellos que sean efectivamente personas, y desechar aquellos que sean empresas, bots, cuentas oficiales de organismos públicos, etc.

Este análisis lo llevaremos a cabo usando la información contenida en el campo “user” de los tuits, y será necesario para ello examinar el texto de la descripción que incorpora, campo que registra la información que los propios usuarios facilitan sobre ellos mismos. Para hacernos una idea del tipo de texto y contenidos a los que nos enfrentamos, hemos dibujado también una nube de palabras que represente esta información.



Figura 4.4.12: Wordcloud de las descripciones de los usuarios.

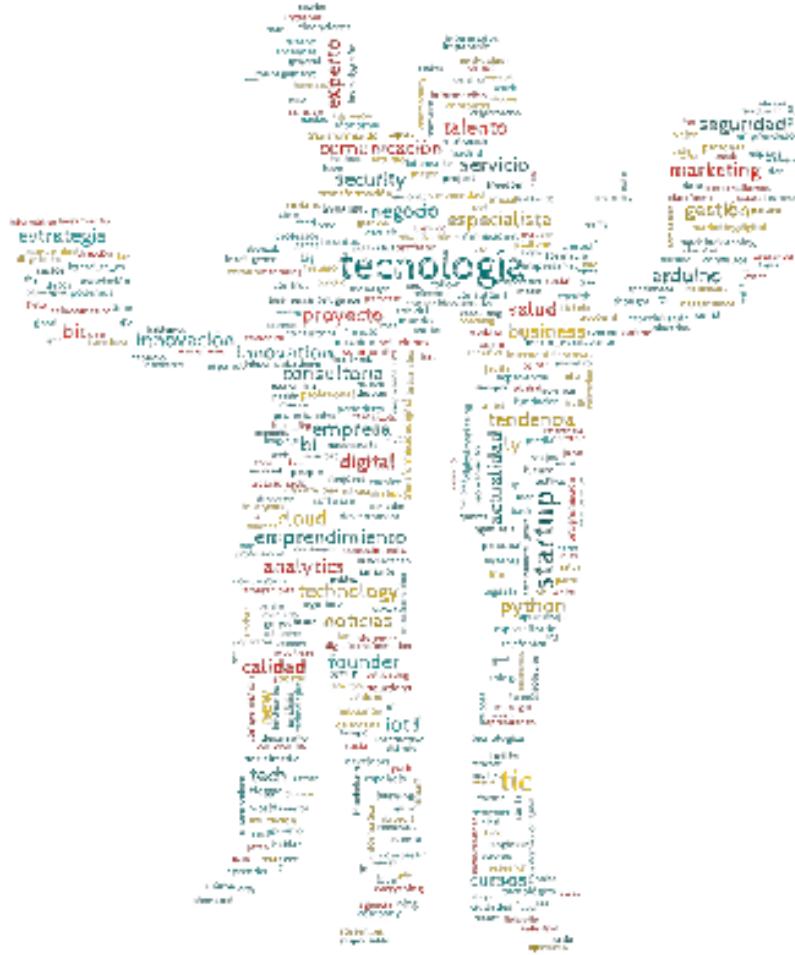


Figura 4.4.13: Wordcloud de las descripciones de los usuarios.

Aparentemente, las descripciones de los usuarios coinciden con los perfiles que nos interesan.

Origen de los tuits

Otro elemento que podría proporcionarnos información relevante para nuestro objetivo es la aplicación de origen del tuit. Un tuit emitido desde un dispositivo móvil es muy probable que haya sido emitido por una persona y no por un bot o una empresa. Entre los campos del tuit, hay uno que refleja esta información: "source"⁴. Este campo describe la utilidad empleada para publicar el tuit y está consignada en formato HTML. Los tuits publicados desde la web de la aplicación tienen un valor en el campo "web".

⁴ <https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/tweet-object>

A continuación describimos los principales orígenes de los tuits almacenados en nuestra base de datos, y al igual que con los hashtags, comparamos los de los tuits descargados, y los de aquellos citados o retuiteados en ellos:

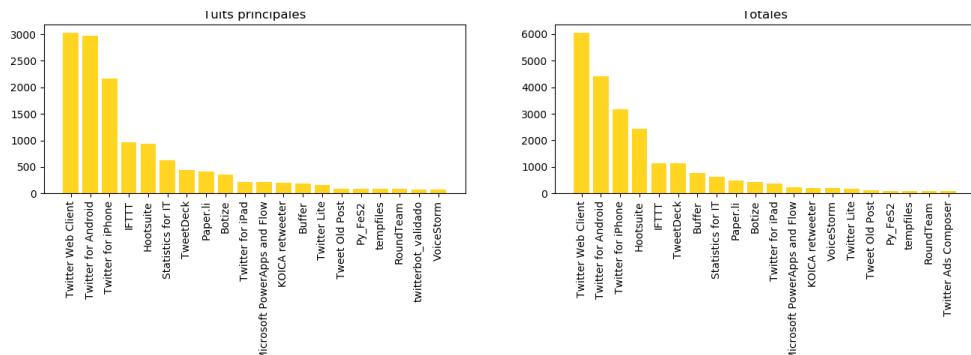


Figura 4.4.14: Orígenes más frecuentes en los tuits descargados (izq.) y en los tuits descargados y citados y retuiteados en ellos.

Estos orígenes tienen las siguientes características:

- Twitter web client: son tuits producidos desde la página web de Twitter
- Twitter for Android, Twitter for iPhone, Twitter for iPad: tuits enviados desde las aplicaciones de Twitter para los sistemas operativos de los dispositivos móviles.
- Hootsuite, Paper.li, RoundTeam: son herramientas que sirven para gestionar la actividad en plataformas sociales, por ejemplo publicar automáticamente contenidos (Twitter entre otras).
- Statistics for IT, Koica retweeter: parecen retuits automáticos por parte de alguna organización.
- TweetDeck es una herramienta oficial creada por Twitter para gestionar y controlar varias cuentas desde un solo lugar. Se pueden controlar notificaciones, menciones, mensajes y la actividad en general de una o varias cuentas, y por supuesto programar envíos de tuits.
- Botize, IFTTT son herramientas para automatizar tareas, en particular en Twitter. Los tuits con este origen serán probablemente de bots.
- PowerApps and Flow: PowerApps es un servicio de Microsoft orientado fundamentalmente a empresas para crear aplicaciones. Microsoft Flow es un servicio también enfocado sobre todo a empresas, para principalmente ayudarles a automatizar tareas entre sus sistemas y algunos de terceros (Twitter entre ellos).
- Twitter Lite: es una versión oficial de Twitter, enfocada sobre todo a usuarios de países emergentes, que funciona a través del explorador y que minimiza el uso de datos y mejora la velocidad de carga en conexiones lentas.

- Tweet Old Post: es un plugin de WordPress, que retuitea de forma aleatoria y automática entradas antiguas de un blog alojado en dicha plataforma.
- Buffer, Py_FeS2, tempfiles, twitterbot_validado: no está muy claro lo que representan. En Twitter, @Py_FeS2 es un usuario nuevo, de Noviembre de 2017, que por el volumen tuiteado parece un bot.
- VoiceStorm: es una aplicación que permite a los empleados de una empresa promocionar la entidad en la que trabajan a través de las redes sociales.

De este análisis parece que si un tuit proviene de un origen como Twitter web client, Twitter for Android, Twitter for iPhone, Twitter for iPad, Twitter Lite, Tweet Old Post y VoiceStorm, es bastante probable que el usuario sea una persona (y por tanto un posible usuario relevante para el objetivo del proyecto). A continuación mostramos los tuits clasificados como “posibles personas” y “posibles otros” atendiendo a lo expuesto anteriormente (en la sección 5.3 volveremos a esta cuestión):

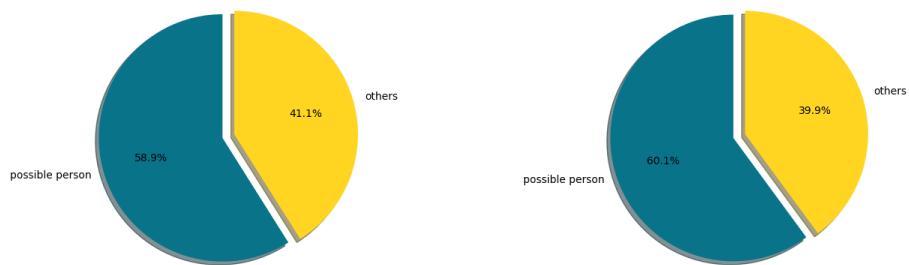


Figura 4.4.15: Porcentaje en los datos de los tuits cuyo origen indican posibles personas en los tuits descargados (izq.) y en los tuits descargados y citados y retuiteados en ellos.

Capítulo 5

Limpieza de datos: extracción de usuarios relevantes

Como hemos visto en el análisis exploratorio inicial de los datos, la fase de limpieza de la información va a ser muy relevante para conseguir nuestro objetivo final. Se trata de extraer, a partir de los tuits almacenados, una lista de usuarios que pudieran constituirse en candidatos adecuados a una oferta de trabajo (como, por ejemplo, las de la figura 1.1.2). Para ello, de todos aquellos usuarios de los que tenemos constancia en los tuits recogidos, hemos de seleccionar aquellos que sean personas (eliminando bots, empresas, etc.) y que hayan publicado contenido relacionado con la materia de referencia, en este caso la ciencia de datos.

Esquemáticamente, el proceso descrito en este capítulo es el siguiente:

1. Consideramos todos los tuits que hemos descargado, tanto los originales como los datos de los tuits que han sido retuiteados o citados en los originales.
2. Estudiamos los idiomas en que han sido escritos, y seleccionamos aquellos que usaremos en el análisis (sección 5.1).
3. Discriminamos los tuits cuyo contenido está relacionado con la ciencia de datos o el Big Data (los “relevantes”, sección 5.2).
4. Consideramos los usuarios que han producido esos tuits de contenido relevante, quitando los repetidos de forma consistente con el resto del proceso (ver detalles en la subsección 5.2.4).
5. De los usuarios relevantes, seleccionamos aquellos que clasificamos como personas (sección 5.3).
6. De aquellos usuarios clasificados como personas, de nuevo eliminamos los duplicados, según el criterio expuesto en la sección 5.3.4.

El resultado de este proceso será entonces una tabla de id de usuarios que procederemos más tarde a ordenar en función de su relevancia.

Los desarrollos descritos en este capítulo están en el fichero `user_selection.py`, que puede encontrarse en el repositorio de GitHub.

5.1 Detección del idioma

El primer paso para poder analizar el contenido de un tuit y determinar si dicho tuit (y por tanto el usuario que lo ha publicado) está relacionado con la ciencia de datos o el big data, es determinar el lenguaje en el que está escrito. Como ya hemos mencionado, aunque la búsqueda en Twitter se realizó solicitando el campo “languages = [“es”]”, obtenemos algunos tuits en otros idiomas, como estos dos que mostramos a continuación:

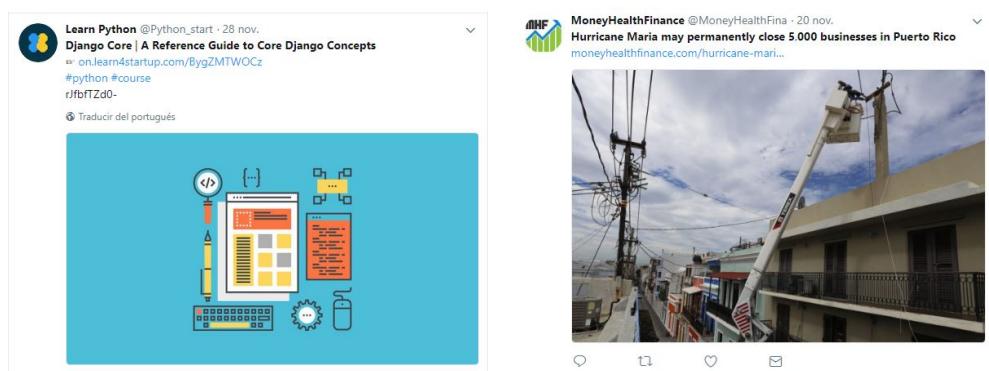


Figura 5.1.1: Tuits no solo en español.

Los primeros trabajos sobre el problema de la detección automática del lenguaje de un texto se remontan a la década de 1970 [9]. En la mayoría de las propuestas, existe una fase de entrenamiento, sobre textos previamente clasificados, en la que se produce un modelo del lenguaje (tal vez uno por lenguaje), y una fase de reconocimiento, en la que el lenguaje de mayor verosimilitud para el texto se extrae a partir de la aplicación de los distintos modelos. La clave de todos estos métodos es la modelización del lenguaje, algo que puede conseguirse atendiendo a diversas características diferenciadoras: fonemas, morfología, sintaxis y/o prosodia.

La aplicación de dichas técnicas a textos provenientes de entornos web, blogs, foros, etc. no está exenta de problemas, ver por ejemplo [10]. Los textos procedentes de entornos web en general, y de Twitter en particular, presentan elevados niveles de lo que podríamos denominar como “ruido”, por ejemplo:

- suelen ser textos cortos, lo que dificulta la aplicación de técnicas basadas en frecuencia de palabras o caracteres,
- presencia de enlaces web, etiquetas, emoticonos y otros caracteres propios del entorno,
- uso de jerga, lenguaje informal y palabras en idiomas distintos del principal del texto,
- modificaciones de la ortografía, que van desde palabras abreviadas (“q”, “xa”) a expresiones enfáticas (“mooooooooooooaaaaaaa”), por poner dos ejemplos.

A pesar de que es posible encontrar corpus de textos con estas características ya clasificados por idioma¹ que pudieran servir para entrenar un modelo que determinara el lenguaje, hemos optado por usar un clasificador que no necesite entrenar un modelo, ya que esta parte del proyecto no es la principal. Hemos encontrado referencias, como [10], que apuntan al buen comportamiento de clasificadores que no depende de conjuntos de entrenamiento (basados en “small words”, también denominadas “stop words”, y trigramas). Pero este método requiere de una implementación *ad-hoc*, y las opciones disponibles, listas para usar, nos han parecido lo suficientemente buenas para el objetivo que perseguimos en este apartado.

En el entorno de Python, hemos encontrado varios paquetes que tratan el problema de detectar el idioma de un texto automáticamente. Los más comentados son los siguientes:

- `langid` [11]: es un paquete que proporciona un clasificador Naïve Bayes de textos, que usa n -gramas (secuencias de n caracteres en el texto, con $1 \leq n \leq 4$). El clasificador está pre-entrenado sobre diversos corpus de texto, en un total de 97 idiomas. Según los resultados explicados en el artículo de presentación del trabajo, es un método con una exactitud (*accuracy*) del 94%.
- `langdetect`: de Nakatani Shuyo, también es un clasificador Naïve Bayes basado en n -gramas, con normalizaciones heurísticas, <https://github.com/shuyo/language-detection>.
- `LDIG`: es un clasificador específico para Twitter, creado por el autor de `lagdetect` para solventar las carencias de éste en la clasificación de mensajes cortos, <https://github.com/shuyo/ldig>. Soporta menos idiomas que los anteriores.
- `equilid`: es el paquete de más reciente creación que hemos encontrado, que además está especialmente diseñado para tratar el “ruido” que comentábamos caracteriza los textos de Twitter. Está concebido para identificar dialectos urbanos y tratar correctamente expresiones de jerga.

Dado el tema que nos ocupa, relacionado con ciencia de datos, la especialización de `equilid` y la dificultad de su instalación en el sistema disponible (usa una versión de TensorFlow que aparentemente no está desarrollada para Windows), nos hace descartarlo de entrada. En [12] se muestra que el comportamiento de cualquiera de los clasificadores restantes es lo suficientemente bueno por separado para el objetivo de este proyecto. Sugieren que combinar dos o más clasificadores puede mejorar la exactitud de la clasificación, y que en general, la limpieza de los tuits (urls, etiquetas, menciones, etc.), si bien mínimamente en algunos casos, suele mejorar el comportamiento de los modelos (excepto en el caso de `langid`). Finalmente, nos hemos decidido por usar el algoritmo del paquete `langid`, visto el buen resultado reportado, y la facilidad de uso del mismo.

En [12] se muestra que, en un contexto general de detección del lenguaje, el paquete `langid` no parece beneficiarse de una limpieza del tuit para retirar urls, menciones, etiquetas y emoticonos

¹https://blog.twitter.com/engineering/en_us/a/2015/evaluating-language-identification-performance.html

del cuerpo del mensaje. Sin embargo, dado que la implementación de esa limpieza no es difícil, hemos comparado la clasificación del lenguaje obtenido con el texto original y con el texto limpio, y hemos observado que en los textos de los 24,128 tuits descargados (contando retuiteados y citados también) hay un 6.37% de tuits que no tienen la misma asignación de lenguaje. Estos tuits suelen ser tuits con pocas palabras o con mucha mezcla de idiomas (generalmente español e inglés).

Como método alternativo, se ha implementado un clasificador manual que solo usa “stop words” para tratar esos casos en los que langid no da una elección clara del idioma. Si el método de los “stop words” da un idioma para el texto que coincide con alguno de los proporcionados por langid (bien el del texto limpio, bien el del texto original), ese será el que se asigne al tuit. Y si no, lo dejaremos clasificado con un idioma desconocido.

Hemos aplicado este método de clasificación del lenguaje a los 24,128 textos de los tuits originales, retuiteados o citados, y resulta una clasificación en 25 idiomas diferentes. Mostramos a continuación un resumen de los resultados del clasificador:

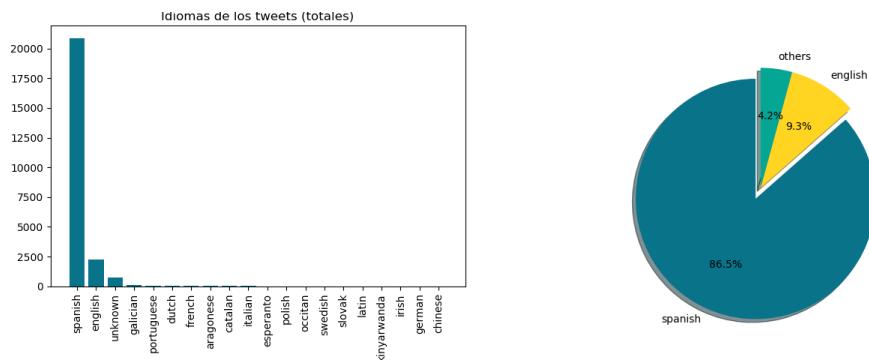


Figura 5.1.2: Clasificación por lenguaje del texto del tuit. Del 4.2% de “others” hay un 2.9% de “unknown”. ¿Errores de clasificación?

Echando un vistazo a los resultados, aquellos que no están clasificados o que están clasificados en idiomas poco probables, digamos, como el esperanto, son aquellos en los que el clasificador seguramente ha fallado. Algunas veces parece ser porque casi no tuvieran texto o porque el texto contenga mucha mezcla de idiomas. Pero otras veces, como en el siguiente caso clasificado en esperanto, no está muy claro por qué:



Figura 5.1.3: Tuit clasificado como esperanto.

Vistos los resultados del clasificador de lenguaje, usaremos para el análisis del contenido de los tuits los idiomas inglés y español.

5.2 Naturaleza del tuit

En esta fase del proyecto el objetivo es discernir si el contenido del texto de un tuit que hemos obtenido de Twitter a través de la búsqueda descrita en la sección 4.2, es efectivamente relevante para seleccionar a su autor como un posible candidato a la oferta de trabajo que nos ocupa.

El criterio para que un tuit tenga un contenido relevante será que describa o haga referencia a alguna de las habilidades o conocimientos requeridos a los candidatos. Es decir, no basta con que el tuit hable de Big Data o de ciencia de datos en general, o de noticias relacionadas con ellos, sino tiene que mostrar algún signo de dichas habilidades y conocimientos.

Para ello, hemos desarrollado un modelo *ad-hoc* que pasamos a describir a continuación.

5.2.1 Construcción del modelo

En este apartado hemos elegido la opción de un **modelo supervisado** con el objetivo de tener más eficiencia en la selección de usuarios relevantes. Por otro lado, la clasificación de los tuits mediante un modelo no supervisado (tipo “bag of words”, por ejemplo), es una técnica que ya exploramos en la clasificación del idioma del tuit que hemos tratado en la sección anterior, y en la clasificación del perfil de usuario que trataremos a continuación.

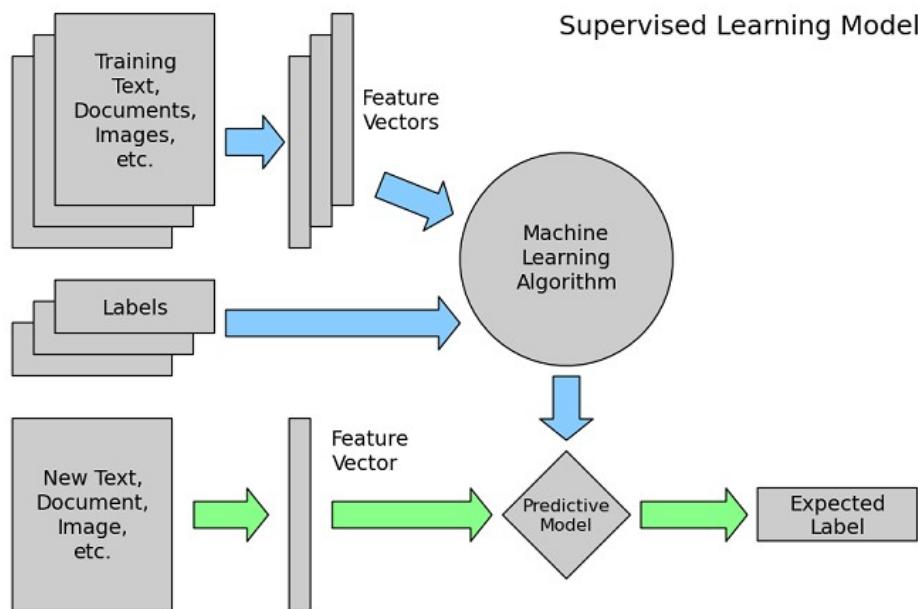


Figura 5.2.1: Proceso de construcción y funcionamiento del modelo para clasificación de textos.

En la figura 5.2.1 se describen esquemáticamente los pasos que hemos llevado a cabo para construir el modelo:

Textos de entrenamiento

Obtuvimos una nueva base de datos de Twitter con 50000 tuits, con una búsqueda distinta a la descrita en la sección 4.2 pero también por palabras clave, a través del API Search de Twitter, y procedimos como en el caso de la base de datos para la extracción de candidatos, a almacenarla en local en una base de MongoDB.

Etiquetado del texto

Trabajamos sobre esa base de datos para construir un data frame que contuviera los textos de los tuits, así como otra información relevante de los tuits (aunque luego no se usa en el modelo), y procedimos a etiquetar manualmente 2000 de esos tuits. Los tuits etiquetados están almacenados en un fichero Excel **Python/relevance_model/ds_sample2.xlsx** de nuestro repositorio de GitHub. Como curiosidad, nos gustaría comentar que al bajar los tuits por palabras claves como Python o SAS, obtuvimos tuits que hablaban de serpientes o de los problemas del Sistema Andaluz de Salud.

Limpieza de los tuits

Antes de entrenar el modelo, intentamos eliminar el “ruido” de los textos sobre los que se va a realizar, sometiéndolos a una limpieza en la que retiramos de los mismos urls, números, signos de puntuación, “stop words”, palabras de menos de dos letras y distinción entre mayúsculas y

minúsculas. La función que lleva a cabo esta limpieza está hecha *ad-hoc*, y puede encontrarse en el archivo **Python/relevance_model/Clasification1.py** de nuestro repositorio de GitHub:

```
def remove_stopwords(text):
    #remove URL
    text = re.sub(r'http\S+', '', text)
    text = re.sub(r'\d{1,9}', '', text)

    # remove punctuation
    regex = re.compile('[' + re.escape(string.punctuation) + ']')

    a=[]
    words = text.split()
    #print(words)
    for i in words:

        new_token = regex.sub(u'',i)
        if not new_token == u'':
            a.append(new_token)

    # remove accentuation
    - - -
```

Figura 5.2.2: Función para limpiar los tuits y entrenar el modelo de clasificación.

Vector de características

Los métodos y algoritmos usados en el entrenamiento y predicción con este modelo son los que nos ofrece el paquete **scikit-learn** de Python.

Los estimadores implementados en **scikit-learn** necesitan transformar las palabras en vectores de características. Para eso elegimos el método **TfidfVectorizer**, porque los textos de tuits son imprevisibles y el algoritmo TF-IDF ajusta la situación de que hay palabras que son más comunes y menos relevantes, además de ser altamente recomendado en análisis de textos por su facilidad de uso y su parametrización.

Este método convierte una colección de textos a una matriz de características TF-IDF. Para ello, crea un “vocabulario” con las palabras que aparecen en los textos y sus características TF-IDF en forma de diccionario. En nuestro contexto, los niveles o características TF-IDF, son números que reflejan la importancia de cada palabra, de forma proporcional a la frecuencia en la que aparece en el documento, pero penalizados por la frecuencia total de ocurrencia de la palabra en todo el corpus. Más concretamente, [35], para cada palabra t y cada documento d :

$$tf(t, d) = \begin{cases} 1 + \log_2(frec(t, d)) & \text{si } freq(t, d) > 0 \\ 0 & \text{en otro caso.} \end{cases}$$

$$idf(t) = \log \frac{|D|}{|D_t|},$$

donde

- $frec(t, d)$ es el número de veces que aparece t en d
 D es la colección de todos los documentos
 D_t es la colección de los documentos que contienen la palabra t
 $|A|$ es el cardinal del conjunto A .

Finalmente, el TF-IDF de un par palabra-documento es

$$tfidf(t, d) = tf(t, d) * idf(t).$$

El resultado del método `TfidfVectorizer` aplicado a un corpus, es una matriz en la que cada documento del corpus es codificado como un vector cuyas entradas son los índices TF-IDF de cada una de las palabras de dicho documento, con respecto a ese mismo documento.

Con esta manera de codificar los textos de los tuits, ya podemos entrenar un modelo de clasificación.

Algoritmo de machine learning

Para entrenar el modelo, dividimos esta muestra codificada en un 30% de datos de test y los restantes datos de entrenamiento, y con esta división procedimos a entrenar el modelo.

Nos hemos decidido por un modelo de tipo Naïve Bayes, que invocaremos a través del modelo `MultinomialNB()` de `scikit-learn`. Después de buscar e investigar, concluimos que este algoritmo, aunque sea uno de los algoritmos más sencillos (lo que también hace que sea más fácil de usar y comprender), tiene muy buen desempeño, velocidad de procesamiento y funciona muy bien en clasificación de textos, por ejemplo al clasificar spam.

Un modelo de clasificación de tipo Naïve Bayes, [36], se basa en asignar a cada observación aquella clase o resultado de la clasificación, a la que es más probable que pertenezca, dados los datos que componen la observación. En concreto, nuestras observaciones están definidas por una serie de variables X_1, X_2, X_3, \dots , y queremos adivinar para cada observación el valor de otra variable Y (discreta y finita). En un modelo Naïve Bayes, el valor de $Y = y_i$ que asignaremos a una observación (x_1, x_2, x_3, \dots) es aquel que hace máxima

$$P(Y = y_i | X_1 = x_1, X_2 = x_2, X_3 = x_3, \dots).$$

Por la fórmula de Bayes,

$$\begin{aligned} & P(Y = y_i | X_1 = x_1, X_2 = x_2, X_3 = x_3, \dots) \\ &= \frac{P(X_1 = x_1, X_2 = x_2, X_3 = x_3, \dots | Y = y_i)P(Y = y_i)}{P(X_1 = x_1, X_2 = x_2, X_3 = x_3, \dots)} \\ &= \frac{P(X_1 = x_1, X_2 = x_2, X_3 = x_3, \dots | Y = y_i)P(Y = y_i)}{\sum_j P(X_1 = x_1, X_2 = x_2, X_3 = x_3, \dots | Y = y_j)P(Y = y_j)} \end{aligned}$$

El cálculo de probabilidades condicionadas en la muestra puede ser muy costoso. El nombre de Naïve en el método viene de la simplificación que el método Naïve Bayes adopta para llevar a cabo estos cálculos, y que consiste en suponer que las variables $X_1 = x_1, X_2 = x_2, X_3 = x_3, \dots$ son independientes entre sí. En ese caso,

$$P(X_1 = x_1, X_2 = x_2, X_3 = x_3, \dots | Y = y_i) = \prod_i P(X_i = x_i | Y = y_i),$$

y el cálculo se aligera notablemente. Sin embargo, esta hipótesis es generalmente la que más problemas nos puede dar con estos modelos, [8], debido a que es muy poco habitual en los problemas reales que las características sobre las que construimos el modelo sean independientes.

Según la documentación de scikit-learn sobre el modelo MultinomialNB()², este modelo en principio debería usarse para la clasificación con características discretas (por ejemplo, conteo de palabras en el caso de clasificación de textos), ya que la distribución multinomial normalmente requiere valores discretos. Sin embargo, en la práctica, conteos no enteros, como el del TF-IDF, también funcionan.

Las siguientes líneas de código son las que hacen el trabajo de entrenamiento del modelo:

```
# Criando um Pipeline - Classificador Composto
# vectorizer/transformer => classifier
text_clf = Pipeline([('tfidf', TfidfVectorizer(ngram_range=(1,1), use_idf=True)),
                     ('clf', MultinomialNB())])
# Fit
text_clf = text_clf.fit(x_train, y_train)
```

Figura 5.2.3: Código para el entrenamiento del modelo de clasificación de relevancia.

Uso del modelo

Una vez entrenado el modelo, lo queremos usar en dos fases: en esta de selección de usuarios y en una posterior para la ordenación de los candidatos (ver sección 6.1). Por ello, hemos guardado el modelo para exportarlo al fichero **Python/relevance_model/modelo_clf.sav** y tener la ocasión posteriormente de importarlo de nuevo. El uso que vamos a hacer del modelo para clasificación es por ello a través de una importación:

```
def procesando_modelo(arquivo,data_set):
    # Cargando el modelo para aplicarlo
    modelo_v1 = pickle.load(open(arquivo,'rb'))

    # Aplicando el modelo en la base final
    # Para aplicar el modelo al campo 'text' deve estar "limpio" igual como fue hecho por aqui. Eso porque
    # al modelo fue entrenado asi...
    c = 0
    for i in range(len(data_set)):
        texto = [data_set['text'][i]]

        pred_x = modelo_v1.predict(texto)
        data_set.set_value(i,'status',pred_x[0])
        if pred_x == 1:
            c = c+1
```

Figura 5.2.4: Código para usar el modelo Naïve Bayes de clasificación de contenidos ya entrenado en **Python/relevance_model/Clasificacion1.py**.

²http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html

Previo a predecir con el modelo, los textos han de pasar por el mismo proceso de limpieza que aquellos textos con los que se entrenó el modelo. Por ello, antes de llamar a esta función, hay que procesar los textos sobre los que vamos a aplicarlo:

```
# *****
# classification if tweet texts topics are relevant for the selection process or not
# *****

arquivo = "C:\\DATOS\\MBIT\\Proyecto\\MBITProject_Data4all\\Python\\relevance_model\\modelo_clf.sav"
all_tweets_clean_texts = pd.DataFrame()
all_tweets_clean_texts["text"] = all_tweets_data["text"].apply(class_model_remove_stopwords)
all_tweets_data["is_relevant_text"] = procesando_modelo(arquivo,all_tweets_clean_texts)
```

Figura 5.2.5: Código para limpiar los textos y posteriormente usar el modelo Naïve Bayes de clasificación de contenidos ya entrenado en **Python/user_selection.py**.

5.2.2 Resultados del modelo de clasificación

		predicted		
		TP	FN	P
actual	n	FP	TN	N
		<i>Confusion matrix</i>		

Figura 5.2.6: Definición de la matriz de confusión.

Para entender mejor como está funcionando el modelo tenemos las métricas de precision, recall, f1-score y la matriz de confusión. Antes, recordemos la terminología que vamos a usar:

TP : “true positive”: observaciones clasificadas correctamente en la clase positiva.

TN “true negative”: observaciones clasificadas correctamente en la clase negativa.

FP “false positive”: observaciones clasificadas incorrectamente en la clase positiva.

FN “false negative”: observaciones clasificadas incorrectamente en la clase negativa.

Con esta matriz de confusión obtenemos las siguientes métricas:

➤ Precisión: mide cómo de preciso es el modelo para detectar positivos, y se define como

$$Pr = \frac{TP}{TP + FP}$$

➤ Recall: mide la sensibilidad del modelo para detectar positivos

$$Rc = \frac{TP}{TP + FN}$$

➤ F1-Score: es una métrica más completa para medir la eficiencia del modelo, y se define como:

$$F1 = \frac{2PrRc}{Pr + Rc}.$$

En nuestro caso, los resultados del entrenamiento son los siguientes:

Matriz de confusión		Métricas de clasificación				
		Precisión	Recall	F1-score	Support	
	0	0.89	0.98	0.93	528	
1	11	0.48	0.15	0.23	72	
0	12	Avg/total	0.84	0.88	0.85	600

Hemos calculado también la curva ROC, obteniendo una exactitud (“accuracy”) de 0.842500526094, y la siguiente curva ROC:

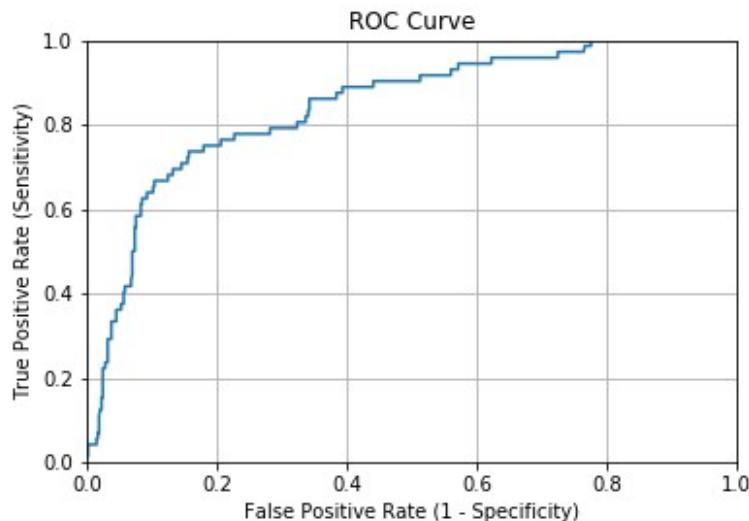


Figura 5.2.7: Curva ROC del modelo de clasificación de textos según su relevancia.

Aceptamos estos primeros resultados y decidimos empezar el proyecto con este modelo para hacer las pruebas hasta el final del proyecto. Sin embargo, estamos seguros de que es posible mejorarlo, principalmente en la precisión de los aciertos en la clase positivas (TP) y en la curva ROC. Hemos empezado a hacer pruebas con otros algoritmos y métodos de optimización (ver sección 8.1).

5.2.3 Resultados de la clasificación del contenido del tuit

De los 24128 tuits analizados, hemos encontrado 365 cuyo contenido hemos clasificado como relevante, lo que representa un 1.51% del total. Los siguientes tuits fueron clasificados como no relevantes:

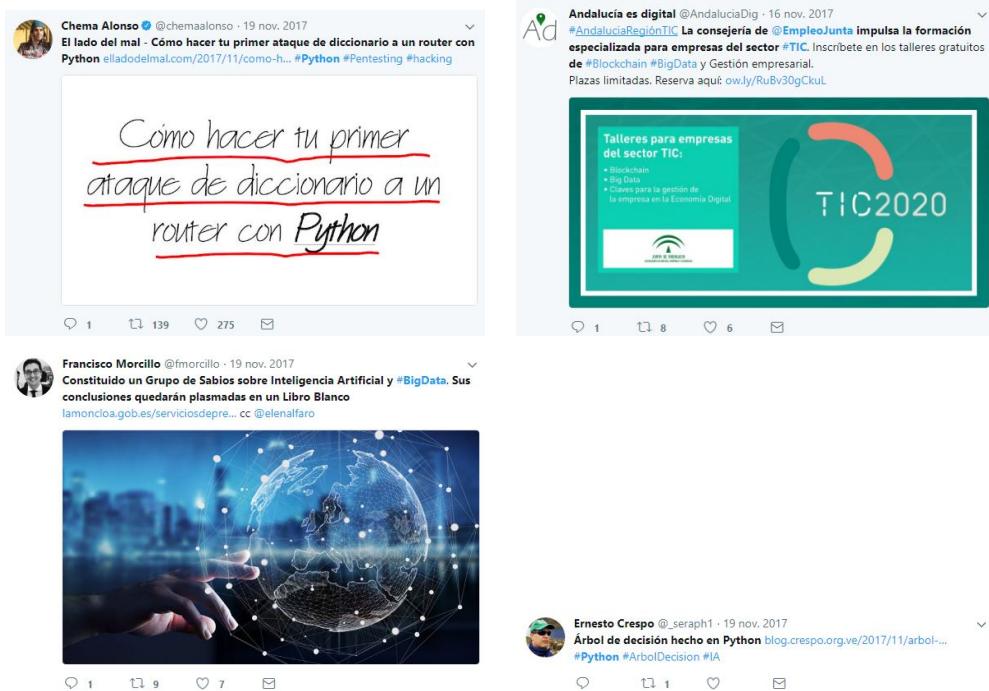


Figura 5.2.8: Tuits no relevantes.

Y los siguientes tuits fueron clasificados como relevantes:



Figura 5.2.9: Tuits relevantes.

5.2.4 Listado de usuarios únicos

El resultado del proceso que acabamos de describir es una lista de tuits clasificados como relevantes o no relevantes para nuestro propósito (encontrar candidatos para un puesto de científico de datos). Nos quedaremos naturalmente solo con aquellos que hayamos clasificado como relevantes.

En esta lista de tuits relevantes, es razonable esperar que haya algunos publicados por el mismo usuario. Para optimizar la ejecución del siguiente tramo del proceso, la selección de aquellos usuarios que son personas, de la lista de usuarios que podemos extraer de aquellos tuits que hemos clasificado como relevantes, nos vamos a quedar solo con instancias únicas de los usuarios.

A lo largo de los procesos descritos en este capítulo y en el capítulo 6, la referencia que vamos a usar para seguir a los usuarios es el número de identificación de usuario de Twitter, al que nos referiremos habitualmente como id. Sin embargo, no es conveniente en esta etapa del proceso, quitar simplemente los id repetidos. Como veremos en la sección 5.3, los campos relevantes para la clasificación de un usuario en persona o no persona, son la url, el nombre de usuario y la descripción del usuario. Todos estos campos pueden ser modificados por el usuario, y podría darse el caso de haber descargado tuits de un mismo usuario, en el que aparecieran diferentes url, nombre de usuario o descripción.

Dado que un usuario podría clasificarse de forma distinta con diferentes url, nombre de usuario y descripción, en esta etapa vamos a considerar que un usuario es un usuario repetido si para dos usuarios sus url, nombres de usuario, descripciones y por supuesto sus id, son iguales.

En el proceso de selección de usuarios, el origen del tuit (el canal o herramienta a través del que se publicó) también va a ser una característica relevante del proceso, pero solo para detectar bots. Al quedarnos con los usuarios únicos, hemos elegido no tomar en consideración el valor de este campo, ya que en principio, un bot siempre publicará desde una fuente automática.

5.2.5 Listado final de usuarios

De los 365 tuits cuyo contenido es relevante, podemos extraer 231 usuarios cuyos campos “user_id”, “url”, “user_name” y “user_bio” no están duplicados. Estos 231 usuarios son los que pasarán la fase de análisis descrita a continuación.

5.3 Tipo de usuario

Para clasificar al usuario respecto a su entidad, y por ejemplo distinguir entre personas, bots y empresas, una de las partes del tuit que más información contiene, a partir de los datos conseguidos, es la descripción que los propios usuarios aportan. Antes de cualquier labor de análisis de esos textos, es necesario saber el idioma en el que están, y por ello de nuevo aplicamos a nuestros datos el identificador de lenguaje que usamos en el apartado anterior.

Primero seleccionamos los datos correspondientes a los usuarios distintos, obteniendo 7,210 usuarios con distinto “id_str”. langid clasifica de forma diferente el idioma de los datos de perfil antes y después de limpiarlos (quitar urls, emojis, hashtags, etc.) en un 5.99% de los casos. Después

de aplicar en éstos últimos el método de las “stop words”, los perfiles de los usuarios han quedado clasificados en 44 idiomas diferentes.

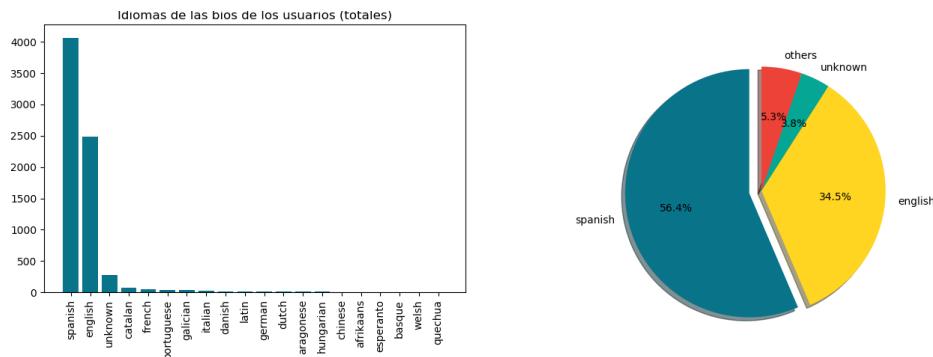


Figura 5.3.1: Clasificación por lenguaje del texto de las bios.

De nuevo, vistos los resultados del clasificador de lenguaje, usaremos para el análisis del tipo de usuario los idiomas inglés y español.

5.3.1 Revisión de la literatura

Para poder hacer una ordenación de los candidatos necesitamos seleccionar a los usuarios o personas reales y tratar dos temas. Por un lado separar a personas y empresas y por otro eliminar los bots o perfiles automatizados.

Bajo el nombre de “empresas”, nos referimos a organizaciones o entidades asociadas con algún objetivo social, político o comercial que, a menudo, tienen una cuenta de Twitter por diversas razones, como la gestión de las relaciones públicas o el servicio al cliente, entre otras. Las organizaciones pueden ser de carácter comercial (por ejemplo, una empresa de marketing) o sin ánimo de lucro (por ejemplo, una ONG) y podrían incluir una empresa, marca, producto o institución benéfica ([21]).

Las personas comunes son usuarios que probablemente estén en Twitter por diversos motivos, como publicar actualizaciones sobre su vida cotidiana, ampliar oportunidades profesionales, mantener contactos con sus amigos y conocidos o descubrir contenido relevante relacionado con sus intereses ([21]).

En Internet, un “bot” (aféresis de robot), “web robot”, “WWW robot” o “Internet bot”, es una aplicación que realiza de forma automática diversas tareas en la red³. En las redes sociales en general, los bots se utilizan para simular la interacción humana, hinchando artificialmente el número de visitas o seguidores, o automatizando respuestas para posicionar mensajes o influir en debates. En Twitter⁴ en particular, un bot se caracteriza por ser una aplicación que controla una cuenta de Twitter a través del API de Twitter. Los bots de Twitter pueden realizar tareas automáticas de publicación de tuits, retuits, seguimiento a otros usuarios, o enviar mensajes directos a otras cuentas. La automatización de cuentas en Twitter se rige por una serie de reglas que distinguen

³https://en.wikipedia.org/wiki/Internet_bot

⁴https://en.wikipedia.org/wiki/Twitter_bot

entre una automatización legítima (difusión de información de interés, publicación de contenidos propios, respuestas automáticas a usuarios, etc.) e ilegítima (violación de los límites de uso del API de Twitter, de la privacidad de otros usuarios, envío de publicidad no deseada, etc.).

Clasificación entre persona y empresa

Hay numerosos ejemplos de literatura que aborda la clasificación del tipo de usuario en Twitter, enfocados a la clasificación en función de la orientación política y étnica ([15], [22]), en función del género ([23], [24], [25]), según sus conexiones sociales ([26]), clasificación en tipos de usuarios como celebridades, bloggers, media y pertenecientes a organizaciones ([26]), etc..

Sin embargo, estos trabajos no se ajustan directamente a nuestro problema de clasificación y requieren muestras previamente etiquetadas. Este enfoque complicaría esta etapa previa a nuestro objetivo fundamental con este trabajo y por tanto las rechazamos.

Otra de las referencias manejadas, [33], un trabajo sobre clasificación de usuarios por la Universidad de Utah de 2014, y a su vez basado en [21], se asemeja más a nuestro objetivo y las técnicas a utilizar son sencillas. En este trabajo proponen basarse en campos como el nombre de usuario, la descripción, la url y el origen de los tweets para clasificar los tuits como publicados por personas y resto. Para ello la idea es la siguiente: se considera que un tuit pertenece a una persona si cumple las siguientes cuatro especificaciones:

E1 En el campo nombre de usuario aparece un nombre que coincide totalmente con un nombre real de persona.

E2 El texto en el campo descripción cumple las siguientes dos condiciones:

- aparecen palabras como “empreendedor”, “consultor”, etc., que determinan que el objeto de la descripción es una persona
- En el campo descripción y en el campo nombre, no aparecen palabras relativas a empresas como “negocio”, “empresa”, etc., que identifican al usuario como una entidad.

E3 Si el campo de url estaba vacío, asignar la categoría de persona y no de empresa (o bot).

E4 Considerar el origen del tweet: las empresas no suelen publicar tweets desde dispositivos de mano.

Detección de bots

La presencia de bots entre los usuarios de Twitter es un asunto no menor, ya que se estima que entre el 9% y el 15% de los perfiles en Twitter son perfiles falsos⁵. El problema de detección de bots en Twitter ha sido un problema muy estudiado a lo largo de los últimos años (ver por ejemplo [18]) y existe numerosa bibliografía al respecto. Parte de la dificultad de este tema es que cada

⁵<http://www.lavanguardia.com/tecnologia/20180203/44478743253/twitter-cuentas-falsas-usuarios-bots-redes-sociales.html>

vez se hace más complicado su identificación, debido a que se camuflan mejor entre los usuarios: criterios que eran válidos hace unos años pueden no resultar tan efectivos hoy.

Una de las soluciones más directas y sencillas es la del servicio propuesto por Botometer (“Bot or not”) que a través de un API proporciona unas probabilidades de que el usuario sea un robot. Esta aplicación funciona tanto para usuarios de lengua inglesa como para usuarios de otros idiomas. Sin embargo, rechazamos esta solución ya que nuestro interés es didáctico.

En [27] se evalúan estrategias de infiltración en Twitter usando 120 bots., mientras que [18] distingue e identifica cuentas de Twitter operadas por tres entidades: humanos, cyborgs y bots. Los autores hacen esta clasificación al observar las diferencias entre las tres entidades en términos de comportamiento, contenido de tuit y propiedades de la cuenta. Sin embargo volvemos a encontrarnos con una metodología demasiado compleja para esta etapa del nuestro proyecto. En el trabajo [20] se muestra que tanto la frecuencia de tuiteo como el origen del tuit son campos muy determinantes para la clasificación de bots. Cabe destacar que el cociente seguidores/amigos no es un buen indicador para la clasificación, ya que aunque previsiblemente los bots no tienen casi amigos, muchos influencers y personas relevantes tampoco, con lo que podríamos eliminar usuarios de interés. Este tipo de metodología para la detección de bots es la que hemos elegido.

5.3.2 Análisis de los datos para la construcción del modelo

En una primera instancia, decidimos entonces aplicar los criterios **E1-E4** para distinguir entre personas y empresas, y usar los indicadores de frecuencia de tuiteo y origen del tuit para distinguir los bots. Al intentar aplicar esta metodologías, observamos lo siguiente:

1. Al inspeccionar el campo “user.name” encontramos ejemplos como los siguientes:

user.name	
Persona	Organización
alvaro solas lara	alidi consultores
manuel bonilla	metric learning
alvaro cerda	metricarts

Figura 5.3.2: Ejemplos de valores en el “user.name”.

Para aplicar el criterio **E1** hemos manejado un archivo con nombres y apellidos de personas en español e inglés, obtenido de estas fuentes:

- https://www.census.gov/genealogy/www/%20data/1990surnames/names_files.html
- http://www.ine.es/dyngs/INEbase/es/operacion.htm?c=Estadistica_C&cid=1254736177009&menu=resultados&idp=1254734710990
- <http://www.buscanombres.com/nombres-gallegos.htm>

➤ <https://script.byu.edu/Pages/Spanish/es/surnames.aspx>

Para el caso de los nombres españoles se han incluido también los específicos para regiones como Cataluña, Comunidad Valenciana y Galicia. A pesar de la amplia lista de nombres y apellidos, en muchos casos solo se identificaba el nombre, en otros solo los apellidos... Con lo que en lugar de validar a las personas con la coincidencia completa de su nombre de usuario, se determinó aceptar que es un nombre válido si se da una coincidencia del 33%. En aquellos casos en los que una persona incluye solo nombre y un apellido, sería suficiente con dejar el corte en el 50%, pero cuando se publican nombre y dos apellidos, es frecuente que solo detectemos el nombre y no los apellidos. De ahí que se tomara la decisión de fijar el umbral en el 33%.

2. Al inspeccionar el campo “user.description” encontramos ejemplos como los siguientes:

user.description	
Persona	Organización
tecnico superior en administracion de sistemas informaticos en red y actualmente estudiante del chee en @tssentinel	consultoria en i+d+i con mas de 10 años de experiencia y un 95% de exito. si tienes una idea innovadora, ponte en contacto en: alidiconsultores@gmail.com
emprendedor abierto a retos. esade. mba executive. dea en umh. ldo en ade ceu. ceo de suma gestion tributaria y pdte de @encuentrosnow http://t.co/5pwii0ffxk	metric arts. business intelligence consulting company
la curiosidad mato al gato... pero el gato murio sabiendo. real madrid. us 1980. ingenieria informatica.	#business development management community #latam. cyber #security senior manager @fmarin_es professional mentoring & #coaching #intelligence security

Figura 5.3.3: Ejemplos de valores en el “user.description” .

Esta inspección, así como la realizada anteriormente a través de las nubes de palabras de las figuras 4.4.12 y 4.4.13, nos sugieren que si aparecen palabras como “emprendedor”, “consultor”, “profesional” o “persona”, el usuario es una persona. Se evitó poner las palabras “I am” o “soy” porque en el caso de algunos bots, en este campo aparecen textos similares a “I am a bot”. Por otro lado, la aparición de palabras como “negocio”, “empresa”, “compañía”, “somos”, “we”... en el campo descripción o en el campo nombre, indicarían que el usuario es una empresa u organización. Se evitaron palabras como “business” por riesgo a que algunos pudieran publicar comentarios del área de *business intelligence*. Tanto en el caso de las palabras que determinan personas, como de las que determinan empresas, tanto en inglés como en español, se crearon sendos ficheros para su almacenamiento.

3. Una primera inspección del campo “url” nos muestra estos ejemplos:

user.url	
Persona	Organización
	https://eventosti.net
http://linkedin.com/in/sebastianmozo	http://cursopic.es
https://github.com/GaryBriceno	http://www.projectbi.net

Figura 5.3.4: Ejemplos de valores en el campo “url”.

En esta inspección descubrimos también que hay más empresas de las que cabría esperar, y fundamentalmente bots, cuyo campo url estaba vacío. No nos parece entonces una buena característica para la clasificación. Sin embargo, sí que se atendió al hecho de que a veces en el campo url el usuario incluye en este campo su perfil de LinkedIn (con lo que el texto “linked” aparece) o su blog personal (en cuyo caso podemos identificar el texto “blog”). Es cierto que en el caso de “blog” se puede incurrir en ciertos errores, dado que también hay bots en cuyo campo url se incluye un blog y es uno de los puntos a mejorar.

4. La idea apuntada sobre el origen de los tuits en la subsección de la página 46, que si el tuit era publicado desde un dispositivo de mano probablemente se trataría de una persona, finalmente resultó que, de nuevo, no era la realidad y esa diferenciación no era tan clara: más organizaciones de las que cabría esperar utilizaban dispositivos de mano para publicar. Por tanto, el usar este campo como criterio único para la clasificación inmediata de personas tal y como proponen en [33], no nos daba buenos resultados. Ellos fundamentalmente lo utilizan para el caso en el que los usuarios no puedan ser clasificados por alguno de los otros criterios debido a que no comparten información sobre su “user.name” o su “user.description”. Sin embargo, en nuestro caso, tan solo el 0.09% de los usuarios (sin repetición) no tienen nada en su campo descripción y un 0.2% en su campo nombre y tan solo 1 que no tiene ni en su campo descripción ni en su campo nombre. Por todo ello, rechazamos el criterio directo sobre el origen.

Con las observaciones anteriores se conseguía una buena clasificación de empresas y de personas. Los fallos en la clasificación de usuarios reales provenían sobre todo de tuits con nombres de usuario extraños o descripciones ambiguas, y fundamentalmente de etiquetar a bots como usuarios, pero no a causa de una mala clasificación entre personas y organizaciones.

Con respecto a las técnicas que queremos explorar para identificar aquellos usuarios que sean bots, nos vamos a fijar en las siguientes características apuntadas en [20]:

- Frecuencia de tuiteo: dado que los bots suelen tuitear mucho, clasificaremos como bots aquellos usuarios cuya frecuencia de publicación sea superior a un cierto umbral. No hay un nivel claro en el que situar dicho umbral. Este sería un punto a estudiar más profundamente y mejorar en un futuro. Nosotros hemos tomado el límite de 350 tuits por día desde la creación de la cuenta (lo que supone un tuit cada dos minutos con doce horas de actividad).
- Origen de los tuits: de acuerdo a los datos mostrados en la figura 4.4.14, se observa que los principales orígenes de los tuits en nuestra muestra son “Twitter Web Client”, “Twitter for Android” y “Twitter for Iphone” (Twitter para dispositivos de mano en los dos últimos casos). Vamos a considerar como bots aquellos tuits producidos desde:

“IFTTT”, “Roundteam”, “Botize”, “Statistics for IT”,
“Koica Retweeter”, “Tweet Old Post”, “Powerapps and Flow”, “Voicestorm”.

Todos ellos están descritos en la página 47.

- Por último, hemos visto que lo más directo es eliminar a aquellos bots que se identifican como tales a través de alguno de los campos “user.name”, “user.description” o “source”. De estos tres campos finalmente descartamos el de “user.description” ya que en el ámbito de la tecnología (al cual pertenecen nuestros perfiles) existen menciones a la palabra “robotica”.

A continuación mostramos algunos ejemplos de datos en nuestra muestra, en los que se aprecian las observaciones anteriores:

user.description	user.name	Source_name
on a kraftwerk noise: "i am a raa-bot.."	kohoraabot	kohoraa_bot
cursos y tutoriales sobre microcontroladores #pic en castellano	microcontrolador pi	botize
proveemos de servicios de #hosting, #housing, servicios en la #nube y servicios de valor anadido sobre infraestructura. especialistas en #sap. info@besh.es	besh	botize
cursos y tutoriales de programacion #python un lenguaje de #programacion moderno, versatil y sencillo. utilzalo tambien con #arduino	programacion python	botize
getting tweaked everyday, to retweet tweets relevant only to python programming.	retweet bot	retweetsapplication
curso de #python para educadores y personas interesadas que quieran iniciarse en el mundo de la programacion. organiza: campus tecnologico virtual.	python en castellano	botize
bot created in node js by @compscidropout created not out of enjoyment but as a	gentooobot	nmc bot
proyectos de #domotica con #arduino	arduino + domotica	botize
todo lo referente a la programacion de #arduino	programacion arduir	botize
herramientas, recursos y formacion para big data [business intelligence & mineria de datos] #bigdata #businessintelligence #datamining	big data en espanol	botize
recursos y tutoriales para arduino/genuino	arduino genuino	botize
ishir is an offshore software development company.	ishir	ishir inc bot
recursos, tutoriales y cursos sobre #python y #linux	linux + python	botize
datos que salvan vidas! almacenamiento, procesamiento y analisis de data medica. videoconsultas, citas, directorio y mas. gratis para latam #esalud #ehealth	siplik	botize

Figura 5.3.5: Ejemplos de valores.

5.3.3 Modelo final

En vista de todas estas observaciones, el algoritmo que hemos usado toma la siguiente forma: un tuit se considera de una persona cuando

- en su campo nombre de usuario existe un 33% o más de coincidencia con la lista de nombres y apellidos,
- y si en su descripción de usuario aparece algún término que identificamos como de persona o no aparecen términos relacionados con empresa ni en la descripción del usuario ni en el nombre de usuario.
- y la frecuencia de publicación de tuits es inferior a 350
- y el origen de tuit no es uno de los que identificamos como automáticos
- y la palabra “bot” no aparece ni en el nombre del usuario ni en el nombre del origen del tuit,
- o bien cuando su url es una url de LinkedIn o de un blog.

El código para llevar a cabo esta limpieza estaría preparado para escalar el modelo a otros casos de uso. Por ejemplo:

- la lista de nombres y apellidos puede extenderse fácilmente para incluir nombres en otros idiomas, si resulta que el texto de las descripciones o de los tuits recolectados se clasifica en otros lenguajes además del inglés y el español, o para extender el alcance de detección del modelo.
- De igual forma, en la lista de términos que sí pueden aparecer en descripción, donde ahora incluimos fundamentalmente vocablos relacionados con perfiles técnicos (consultor, matemático, ingeniero, etc.) para clasificar los usuarios como personas, se puede modificar fácilmente para incluir aquellos que estén relacionados con el perfil requerido en la oferta laboral y mejorar la clasificación.
- Respecto a la lista de términos que no deben aparecer en el campo descripción (porque identifican empresas o bots) se podría utilizar sin ningún cambio. En el caso en el que los tuits sean en otros idiomas diferentes al inglés o al español se podría incluir fácilmente la versión en los idiomas necesarios de las palabras consideradas.
- La lista de aplicaciones identificadas como productoras de tuits automatizados también está consignada en un fichero fácilmente modificable.

5.3.4 Eliminación de duplicados

Una vez obtenida la clasificación en persona o no persona de los usuarios llevada a cabo como hemos descrito, solamente nos queda producir el listado definitivo de candidatos seleccionados (ya no va a haber ninguna clasificación posterior). Esto implica, en primer lugar, que eliminaremos de la lista aquellos usuarios que no han sido clasificados como persona. A continuación, nos interesa extraer las personas únicas que hemos detectado: del listado extraído del paso anterior, sección 5.2.4, podría ocurrir que un usuario fuera clasificado como persona con un conjunto de url, nombre

de usuario y descripción, pero no lo fuera con otro. En esta fase, por tanto, nos limitaremos a eliminar del listado de los clasificados como personas aquellos usuarios con un id repetido.

5.3.5 Resultados de la selección

Aplicando esta metodología a la lista de usuarios únicos obtenidos tras la aplicación de los algoritmos descritos en la sección 5.2, hemos obtenido los siguientes datos: de 231 usuarios únicos analizados, hemos encontrado 155 que clasificamos como personas, lo que representa un 67.1% del total, y de éstos no había ninguno cuyo “user_id” estuviera duplicado. Los resultados relativos están consignados en la siguiente gráfica:

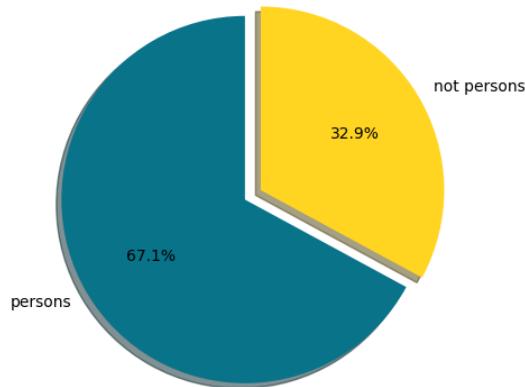


Figura 5.3.6: Porcentaje de usuarios clasificados como personas.

Los siguientes usuarios fueron clasificados como no personas:

PiperLab
@PiperLab_es
Pegados a la actualidad del mundo del #datascience, el análisis, el #bigdata y los modelos...pero con los pies sobre el negocio.

BIG DATA
@SaludBigData
Portal Español de Noticias y Proyectos sobre #BigData en #Salud

Sistemas UCP
@SistemasCuenca

Coding Flux
@CodingFlux_
I retweet everything about #Coding and #SoftwareDevelopment !

Quonext Tourism
@QuonextTourism
Quonext Turismo: más de 20 años de experiencia en desarrollos e implantaciones de software de gestión para el sector turísticos y hotelero

Startup Burgos
@StartupBurgos
Laboratorio del Emprendimiento de Burgos #Burgos #emprendedores #emprendimiento #LeanStartup #StartUp #mentoring

Patroklo™
@Patroklin
La necesidad es el edulcorante de la creatividad

Fernando Cuenca
@fernandocuenca
#BusinessIntelligence #Qlik #QlikSense #Analytics #BigData

Andy Sugden
@AndySugs
Father ov 3 who supports LFC & likes 2 do odd T-SQL query. Buzz off SQL BI. House tunes, reading news & airing me views #SQLServer #PowerBI ...

Figura 5.3.7: Usuarios no clasificados como persona.

Y los siguientes son algunos de los que se clasificaron como persona:

Samuel Viana
@digifish
Self-made. Curious. Want to have a bird view of things. Biology, Informatics and Science Fiction. Lion. Em português prefiro factos em vez de fatos.

king in the north
@Htivan95
Un buen friki de los videojuegos , la buena música ademas de viodiado a las películas.series y risas.Informático y futuro programador

Appsco Cali
@AppscoCali
Fase de Crecimiento y Consolidación #TIC de @AppsCo y @Ministerio_TIC, operado por @CamaratCali, en el #Coworking @PluralDNA, #StartUp ...

CAPITAL RADIO
@CAPITALRADIOB
Eres líder cuando otr@s te imitan. En Madrid ahora 105.7 FM. ¡Vente!

Figura 5.3.8: Usuarios clasificados como persona.

En el siguiente capítulo explicamos cómo ordenamos estos 155 usuarios.

Capítulo 6

Modelado de los datos: ordenación de los usuarios

Una vez que ya tenemos una lista de usuarios cuyas publicaciones en Twitter indican que podrían ser candidatos para las ofertas de trabajo, el siguiente paso es ordenarlos de acuerdo a su relevancia. A continuación describimos brevemente los dos criterios que usaremos para ordenarlos (el detalle de los algoritmos y procedimientos llevados a cabo en cada caso se hará en la secciones siguientes):

- Ordenación de tipo índice h . El índice h es un sistema propuesto por Jorge Hirsch, de la Universidad de California, para la medición de la calidad profesional de los científicos, en función de la cantidad de citas que han recibido sus artículos científicos¹. Un científico tiene índice h si ha publicado h trabajos con al menos h citas cada uno. El índice se diseñó para medir eficazmente la calidad del investigador, a diferencia de sistemas de medición más sencillos que cuentan citas o publicaciones, donde se hace una distinción entre aquellos investigadores que tienen una gran influencia en el mundo científico de aquellos que simplemente publican muchos trabajos. En nuestro campo, lo que queremos es medir la relevancia de los candidatos, así que una métrica de este tipo aplicada a los tuits publicados de la temática de interés (Big Data y ciencia de datos en nuestro caso), nos ayudará a determinarla. Los detalles, en la sección 6.1.
- Ordenación en función del papel de cada usuario dentro de la red de usuarios seleccionados (sección 6.2). Para ello, construiremos el grafo de los candidatos y sus relaciones, siendo estas relaciones dirigidas (A está relacionado con B si A sigue a B).

Con estos dos métodos pensamos que quedará bastante bien representado el panorama de candidatos a partir de la información extraída de Twitter. Todos los cálculos descritos en esta parte del proyecto están en el archivo `user_rank.py` del repositorio de GitHub.

En las siguientes secciones detallamos los algoritmos que hemos usado para cada sistema de clasificación, aportando la justificación de las diversas elecciones que hemos llevado a cabo.

¹<https://en.wikipedia.org/wiki/H-index>

6.1 Índice h

La particularización del índice h a nuestro contexto podría ser del siguiente modo: un usuario de Twitter tendrá índice h si h de sus N tuits sobre un determinado tema, han sido retuiteados al menos h veces.

Calcular el índice h implica por tanto los siguientes pasos:

1. Explorar el timeline² de cada usuario. Para ello usaremos la función `user_timeline` que ofrece el paquete de Python Tweepy.
2. Determinar, dentro de ese timeline, qué tuits son del tema que nos interesa (Big Data o ciencia de datos) y descartar el resto.
3. De cada tuit publicado sobre el tema de interés, anotar su número de retuits y calcular el índice h .

La descarga del timeline de usuarios tienen varias limitaciones en el acceso a través del API de Twitter³:

- límite de 1500 llamadas por cada ventana de 15 minutos,
- solo se permite acceder a los 3200 más recientes tuits de cada usuario,
- el número de tuits descargados en cada llamada al API solo puede ascender a 200.

Para que el proceso de obtención de datos no se pare cuando alcanzamos el límite de llamadas (da un error), la función de Tweepy permite incluir el parámetro `wait_on_rate_limit = True` para gestionar ese error y esperar a reanudar el proceso cuando sea posible.

La función del API de Twitter para obtener el timeline de los usuarios no tiene la opción de buscar tuits de forma temporal (por ejemplo, los de los tres últimos meses). Hemos optado por evaluar el timeline respecto a los últimos 200 tuits publicados por parecernos un número lo suficientemente amplio para el objetivo de clasificar los usuarios, si bien se podría incluir en el código la gestión de un número mayor de tuits en el timeline (aumentando el número de llamadas al API).

Un aspecto importante a la hora de evaluar el impacto del usuario es distinguir entre sus publicaciones originales y los retuits. De los tuits que componen el timeline, vamos a extraer la siguiente información:

- el número de tuits descargados(que en principio serán 200, pero podrían ser menos),
- el número de tuits sobre el tema de interés, y por tanto el porcentaje de tuits de interés sobre el total de tuits evaluados,

² "Timeline" es la palabra que usan en Twitter para referirse a un flujo de publicaciones a lo largo del tiempo. El timeline de un usuario es el flujo de las publicaciones de ese usuario.

³https://developer.twitter.com/en/docs/tweets/timelines/api-reference/get-statuses-user_timeline

- sobre los tuits de interés, el porcentaje de aquellos que son retuits,
- el índice h de los tuits originales. Para calcular el índice h contaremos tanto las veces que se ha retuiteado el tuit como las que ha sido citado.

Como información adicional también obtendremos el idioma en el que los usuarios han publicado los tuits. En esta parte del proyecto reutilizamos por tanto dos herramientas mencionadas en la parte de selección de usuarios: el modelo que nos permite detectar si un tuit es relevante o no (explicado en la sección 5.2) y el algoritmo para detectar el idioma de un texto (visto en la sección 5.1).

Al intentar bajar el timeline de algunos usuarios se pueden obtener diversos mensajes de error (por ejemplo, porque el usuario tenga protegido su timeline, porque el usuario haya dejado de existir, etc.) de forma que el proceso puede fallar. El código debe contemplar ese caso y permitir que la ejecución continue con el resto de usuarios:

```
errors = []
for i in range(len(df2["user_id"])):
    user_id = df2["user_id"][i]
    print("Processing user number "+str(i)+" user_id = "+str(user_id))
    # timeline extraction: caring for errors for protected user timelines
    try:
        user_cursor = api.user_timeline(user_id = user_id,
                                         screen_name = df2["user_screenname"][i],
                                         count = n_tuits)
        errors.append("")
    except tweepy.TweepError as e:
        print ("Failed to download user " + str(user_id) + " timeline")
        print ("Exception: " + str(e))
        print ("Skipping... ")
        user_cursor = []
        errors.append(e)
```

Figura 6.1.1: Control de errores en la descarga de timelines.

Una vez obtenidos los tuits relevantes del timeline de cada usuario, y el número de veces que han sido retuiteados, la función que calcula el índice h es la siguiente:

```
def get_h_index (n_retweets):
    # needs a vector with the citations (number of retweets) of each published tweet
    nr = sorted(n_retweets, reverse = True)
    h = 0
    for i in range(1, len(nr)+1):
        if nr[i-1]>i : h=i
        else: break
    return h
```

Figura 6.1.2: Función para el cálculo del índice h .

6.2 Grafo relacional

Nuestro objetivo en esta parte es explorar las relaciones entre los usuarios que hemos identificado como relevantes. De forma natural, como en cualquier red social, se pueden definir multitud de estructuras de tipo grafo para describirlas. Nosotros vamos a usar un grafo cuyos vértices serán los usuarios, y cuyas aristas o arcos serán las relaciones entre ellos, de tipo dirigido: el usuario A está relacionado con el usuario B si A sigue a B (y puede muy bien ocurrir que B no esté relacionado con A, según esta definición).

Para esta parte necesitaremos entonces descargar de Twitter la información necesaria para construir el grafo de relaciones: los seguidores (“followers”⁴) de cada uno de los usuarios identificados como potenciales candidatos. En principio, podríamos usar tanto los followers como los friends de cada uno de los usuarios de nuestra lista. Pero en realidad no hace falta, ya que si el usuario A es un friend del usuario B, B es un follower del usuario A, y lo consideraremos al estudiar los followers de A.

Para explorar los followers de cada usuario seleccionado usaremos la función `followers_ids` del paquete Tweepy. Esta función es una interfaz para la función del API de Twitter que permiten acceder a la lista de followers. Esta función del API de Twitter tiene una limitación de 15 llamadas al API por cada ventana de 15 minutos⁵, con lo que la gestión de los límites es especialmente importante en este caso. El parámetro `wait_on_rate_limit = True` al crear el acceso al API a través de Tweepy también nos va a ser de utilidad.

Por otro lado, la información que puede descargarse con esa función también está limitada en cantidad: el número de followers que puede obtenerse en cada llamada al API es a lo sumo 5000. Igual que antes, usando un código con varias llamadas al API, podríamos obtener un número más elevado de seguidores.

Nuestro grafo será por tanto un par ordenado $G = (N, g)$, donde N es un conjunto de vértices o nodos, que serán los usuarios identificados como potenciales candidatos para la oferta de trabajo, y g es un conjunto de aristas o arcos, descritos como pares de nodos ordenados:

$$g = \{(a, b) : a, b \in N, a \text{ sigue a } b\}.$$

En las representaciones de un grafo dirigido, una relación (a, b) se suele describir con una flecha que sale de a y apunta a b . Al extraer la información sobre los usuarios hemos de quedarnos con los followers de cada uno que están en el conjunto de usuarios, es decir, solo nos quedamos con las relaciones entre candidatos:

```
def get_followers_ids(user_id, user_ids_set, api):
    ids = []
    page_count = 0
    for page in tweepy.Cursor(api.followers_ids, id=user_id, count=5000).pages():
        page_count += 1
        # print ("Getting page " +str(page_count)+" for followers ids " + str(user_id))
        ids.extend(set(map(str, page)) & user_ids_set)
    return ids
```

Figura 6.2.1: Followers y relación entre usuarios.

Una vez obtenidas estas relaciones, vamos a necesitar una librería para procesar el grafo y sacar las pertinentes conclusiones. A este respecto, tenemos varias opciones que considerar. Entre otras:

- **Graph-tool**: los algoritmos están implementadas en una librería C++, lo que aporta mucha

⁴A aquellos usuarios que siguen a un determinado usuario, en Twitter se les denomina “followers”. Y se llaman “friends” aquellos a los que dicho usuario determinado sigue.

⁵<https://developer.twitter.com/en/docs/accounts-and-users/follow-search-get-users/api-reference/get-followers-ids>

rapidez en la ejecución. Es buena procesando grafos grandes. La instalación en entornos Windows no está soportada⁶.

- NetworkX es fácil de usar, para conjuntos de datos pequeños. Está bien documentado. Es compatible con Python 2.7, 3.4, 3.5 y 3.6.
- SNAP (Stanford Network Analysis Platform): es un sistema para el análisis y la manipulación de grandes redes. Está implementada en C++, con una interfaz en Python. Disponible para Python 2.7⁷.
- igraph: es una colección de herramientas para análisis de grafos, con interfaces en R, Python y C/C++. Compatible con Python 2.6, 2.7 y 3.2.
- APGL (Another Python Graph Library): es una librería sencilla para procesar grafos, disponible en principio para Python 2.7⁸.

Vistas las opciones, nos hemos decidido por usar NetworkX. Esta herramienta nos va a permitir calcular varias métricas que usaremos para estudiar el papel de cada usuario dentro de la red y medir su relevancia ([29], <https://www.sci.unich.it/~francesc/teaching/network/>), y también explorar la estructura de la red, como exponemos a continuación.

6.2.1 El papel de los usuarios: medidas de centralidad

Las medidas de centralidad de los nodos de un grafo tratan de responder a la cuestión de cuáles son los vértices más importantes o centrales dentro del grafo. La “importancia” de un nodo puede ser entendida de múltiples formas, y en consonancia hay diversas medidas de la misma.

Una vez que hemos determinado la medida de centralidad que nos interese para los nodos, podremos usarla para ordenar la lista de los nodos (que es nuestro objetivo principal). Todas las medidas de centralidad que vamos a describir a continuación se pueden calcular con las funciones que proporciona el paquete NetworkX.

Centralidad por grado

El grado (degree) de un nodo n es el número de aristas incidentes en n . En los grafos dirigidos podemos distinguir entre in-degree (número de aristas que apuntan a n , en nuestro caso el número de followers) y out-degree (número de aristas que parten de n , en nuestro caso, el número de usuarios a los que sigue n). El in-degree representaría la capacidad de “liderazgo” del usuario (a más followers, en principio más liderazgo), y el out-degree representaría el interés del usuario en seguir a otros usuarios, que podríamos interpretar como su interés por el clima general del sector.

⁶<https://git.skewed.de/count0/graph-tool/wikis/installation-instructions>

⁷<https://snap.stanford.edu/snappy/index.html>

⁸<https://pythonhosted.org/apgl/>

Centralidad por autovector (eigenvector centrality)

Es una extensión del concepto de centralidad por grado que captura la intuición de que la importancia de un nodo en la red incrementa por el hecho de estar conectado a otros nodos a su vez relevantes. En un grafo dirigido, como el nuestro, parece natural pensar que es más relevante la importancia de los usuarios que siguen a uno dado, y que por tanto deberíamos usar los enlaces que apuntan al usuario en cuestión (las aristas que cuentan para el in-degree) para medir su propia importancia.

Sea c el vector columna de las centralidades por autovector de los N nodos del grafo y sea A la matriz $N \times N$ de adyacencia del grafo, en la que a_{ij} representa la contribución del nodo n_i al “prestigio” del nodo n_j (en nuestro caso, $a_{ij} = 1$ si n_i sigue a n_j y cero en caso contrario). La idea detrás de esta centralidad ([31]) es que la centralidad de un nodo será la suma de las centralidades de los nodos que le apuntan, ponderadas por el peso de la unión, esto es:

$$c_i = \sum_j a_{ji} c_j$$

que en notación matricial es

$$c = A^T c.$$

Para asegurarnos de que esta ecuación tenga solución debemos considerar una formulación más general, en la que la centralidad no es exactamente la suma de las centralidades de los nodos que contribuyen, sino solo un múltiplo de dicha suma:

$$\lambda c = A^T c,$$

de tal forma que c , el vector de las centralidades del grafo es un autovector de la matriz A^T . Si A es una matriz $N \times N$ hay en general N soluciones a dicha ecuación, correspondientes a los N autovalores de A^T . Suele considerarse como solución el autovector correspondiente al máximo autovalor.

La centralidad por autovector tiene un problema cuando alguno de los nodos no tiene enlaces que lo apunten (es decir, si alguno de los usuarios no tuviera ningún seguidor entre los demás usuarios, aunque él sí que siguiera a otros), ya que en ese caso un nodo así siempre contribuirá cero, y por tanto, en algunos casos la centralidad por autovector de todos los nodos sería cero:

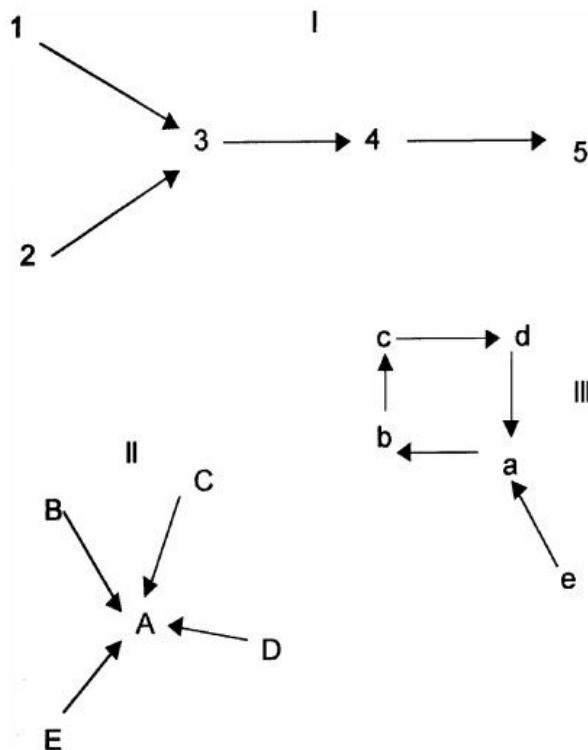


Figura 6.2.2: En los grafos I y II de estos ejemplos (extraídos de la página 192 de [30]), la centralidad por autovector de todos los nodos es 0.

Una medida de centralidad que puede ayudar en este tipo de situaciones es la centralidad de Bonacich.

Centralidad de Bonacich

Para paliar el problema mencionado anteriormente de la centralidad por autovector, podemos usar la centralidad de Bonacich, que parte de la siguiente idea: todos los nodos tienen una centralidad mínima. Con la misma notación que en el apartado anterior, la centralidad de Bonacich puede definirse (ecuación (5) en [30]) como:

$$c_i = e_i + \alpha \sum_j a_{ji} c_j, \text{ que en notación matricial es } c = e + \alpha A^T c$$

donde $\alpha \in [0, 1]$ y e es un vector constante. Esto es, la centralidad de cada nodo es la suma de una centralidad mínima (o exógena, como la denominan en [30]) y de las contribuciones a la centralidad de las centralidades de los nodos que apuntan a dicho nodo. Esta contribución se amortigua a través del parámetro α , ya que para nodos a distancia k de un nodo dado, su contribución es múltiplo de α^k . Si $\alpha = 0$, no hay transmisión de centralidad.

PageRank

En las centralidades por autovector y de Bonacich, los nodos distribuyen de forma indiscriminada su centralidad a todos los demás nodos a los que apunten, por lo que un nodo importante que apunta a muchos vecinos hace importantes a todos ellos. La idea de PageRank es repartir la centralidad aportada por un nodo entre todos aquellos a los que apunta, de tal forma que cada uno de ellos recibirá solo la parte proporcional de la centralidad del nodo de partida, esto es, la centralidad del nodo dividida por su out-degree (número de enlaces salientes):

$$c_i = (1 - d) + d \sum_j a_{ji} \frac{c_j}{od(n_j)},$$

donde $od(n_j)$ es el out-degree del nodo n_j , y d es un factor de amortiguación que tiene un valor entre 0 y 1. En la aplicación que hace Google de este algoritmo (tienen una patente sobre él), parece que $d = 0.85$, aunque la implementación concreta del algoritmo es por supuesto secreta. El papel de d es similar al de α en la centralidad de Bonacich, en cuanto a la transmisión de la centralidad, y además ayuda a la convergencia del cálculo del PageRank (se resuelve de forma iterativa). En [34] se puede encontrar una deliciosa explicación del algoritmo.

Cercanía (closedness centrality)

En un grafo conexo, la cercanía de un nodo es el inverso de la suma de las longitudes de los caminos mínimos que unen dicho nodo con todos los demás:

$$c_i = \frac{1}{\sum_j d(n_j, n_i)},$$

donde $d(n_j, n_i)$ es la distancia geodésica⁹ entre los nodos n_j y n_i .

Cuanto más central es un nodo, más cerca está del resto de nodos. Tales vértices pueden tener mejor acceso a la información de otros nodos, o una influencia más directa en ellos. Por ejemplo, una persona con mayor cercanía puede ver cómo sus opiniones alcanzan a otros usuarios más deprisa que las de alguien con menor cercanía.

Frecuentemente, en lugar de la suma se considera la media de los caminos más cortos, de tal forma que se pueden comparar grafos de diferentes tamaños. Entonces, la formulación es:

$$c_i = \frac{N - 1}{\sum_{j \neq i} d(n_j, n_i)},$$

donde N es el número de nodos en el grafo. Si el grafo es muy grande, N y $N - 1$ son muy parecidos y suele usarse:

$$c_i = \frac{N}{\sum_j d(n_j, n_i)}.$$

Considerar las distancias a o desde los otros nodos es irrelevante en grafos no dirigidos. Pero en el caso de grafos dirigidos, puede producir resultados totalmente diferentes. Cuando el grafo no es

⁹Las geodésicas son el camino más corto entre dos puntos y no son necesariamente únicas.

conexo, se puede usar la suma de los inversos de las distancias en lugar del inverso de la suma de las distancias, con la convención $1/\infty = 0$:

$$c_i = \sum_{j \neq i} \frac{1}{d(n_j, n_i)}.$$

Intermediación (betweenness centrality)

Captura en qué medida un nodo está situado en los caminos que unen a otros nodos. Los nodos con una intermediación alta pueden tener mucha influencia en la red, dado que “controlan” la información que se transmite entre otros nodos. También son nodos que si desaparecieran impedirían la transmisión de la información y aumentaría la desconexión de la red.

Matemáticamente, sea $n_{k,j}$ el número total de geodésicas que van desde n_k a n_j , y $n_{k,j}^i$ el número de ellas que pasan por n_i . Entonces, la intermediación del nodo n_i es:

$$c_i = \sum_{k,j} w_{k,j}^i = \sum_{k,j} \frac{n_{k,j}^i}{n_{k,j}},$$

donde el cociente $w_{k,j}^i$ expresa lo “en medio” que está n_i de n_k y n_j , y por convención $w_{k,j}^i = 0$ si $n_{k,j} = 0$.

Observemos que la definición cuenta de forma distinta los caminos de n_k a n_j y de n_j a n_k (importante en grafos dirigidos), aquellos que empiezan o terminan en n_i ($n_k = n_i$ o $n_j = n_i$) y también aquellos en los que $n_k = n_j$. Esto atiende al hecho de que parece razonable considerar que un vértice está en medio cuando está en un camino entre él y algún otro, ya que en esos casos, también controla el flujo de información. En cualquier caso, usar otra definición no suele implicar mucha diferencia, ya que en general nos interesarán los valores relativos de la medida, y no su valor absoluto. En ese sentido, suele normalizarse por el número total de pares ordenados del grafo (N^2), para que la medida nos dé un número entre 0 y 1.

La tabla 6.1 resume las medidas de centralidad que vamos a incluir en nuestro análisis y una pequeña idea de su significado.

6.2.2 La estructura de la red

La teoría de grafos también nos permite extraer información muy interesante sobre la estructura de la red de usuarios: hay unas ciertas características que suelen aparecer en estructuras del tipo que nos interesa, patrones que tienen un efecto importante en cómo se comportan estas redes. Esta información no es directamente relevante para la ordenación de los usuarios, pero nos permitirá hacernos una idea más clara de cómo se organiza. Veamos algunas de dichas características.

Centralidad	Descripción
Por grado	Conexión total del usuario con la red
In-degree	Número de followers
Out-degree	Número de seguidos
Por autovalor	Importancia de los seguidores
Bonacich	Importancia de los seguidores más centralidad mínima
PageRank	Importancia proporcional de los seguidores más centralidad mínima
Cercanía	Facilidad de acceso a otros usuarios
Intermediación	Influencia y conexión de la red

Tabla 6.1: Medidas de centralidad e idea de su significado.

Estructura del grafo

- Componentes conexas: una componente conexa de un grafo no dirigido es un conjunto de nodos maximal (que no se pueden añadir más) tal que cualquier par de nodos está conectado. Las componentes conexas son una partición del grafo (son disjuntas y su unión es el grafo completo). En numerosos ejemplos reales, suele aparecer una gran componente conexa (la componente gigante) que cuenta con una mayoría de los nodos (típicamente más del 50%, pero no es infrecuente el 90%). En los grafos dirigidos la noción de conectividad se complica y tenemos componentes débilmente conexas y fuertemente conexas:
 - Dos nodos están en la misma componente débilmente conexa si están conectados por un camino, donde los caminos pueden recorrerse en cualquier sentido a lo largo de cualquier arista (esto es, como si el grafo fuera no dirigido). Se comporta igual que en el caso no dirigido, suele haber una componente gigante en la que se agrupan un alto porcentaje de los nodos.
 - Dos nodos están en la misma componente fuertemente conexa si desde cada uno de ellos se puede ir al otro, a través de caminos dirigidos. Las componentes fuertemente conexas también forman una partición del grafo, y también se puede presentar una componente gigante, aunque debido a las restricciones del grafo, no suele ser tan grande.
- Caminos mínimos (geodésicas)
 - La excentricidad de un nodo en un grafo conexo es la mayor distancia geodésica entre ese nodo y cualquier otro vértice del grafo. En un grafo desconexo, o bien todos los nodos se definen con una excentricidad infinita, o bien se calcula la excentricidad en la componente conexa a la que pertenezca el nodo.
 - Diámetro: la mayor distancia geodésica dentro del grafo (o componente si no es conexo). Es la máxima excentricidad.
 - Radio: la mínima distancia geodésica dentro del grafo.

- Longitud del camino medio: media de todos los caminos mínimos entre los nodos del grafo (menos sensible a valores extremos).
- Distribución del grado de los nodos: del grado, out-degree o in-degree, da una idea de lo conectada que está la red. La distribución conjunta del in y el out-degree nos proporciona información sobre la correlación entre ambas cantidades. En muchos casos, estas distribuciones suelen ser muy asimétricas: muchos nodos con valores bajos, y unos pocos con valores muy altos.
- Transitividad: Tres nodos n_1, n_2, n_3 tales que n_2 está relacionado con n_1 y n_3 forman un triángulo centrado en n_2 . El triángulo es cerrado si n_1 está relacionado con n_3 , y abierto en caso contrario. La transitividad mide la capacidad de la red para que los triángulos sean cerrados, esto es, si un nodo n_1 está conectado con otro nodo n_2 , y éste con un tercero n_3 , entonces n_1 también está conectado con n_3 . El coeficiente de transitividad de una red, también conocido como coeficiente de clusterización, es el cociente entre el número de triángulos cerrados en la red, respecto al número de posibles triángulos. Las redes sociales tienden a tener valores altos de transitividad¹⁰. En las redes dirigidas, la transitividad se calcula como para las no dirigidas, ignorando la dirección de las aristas.

Grupos de nodos

Desde el punto de vista de nuestra aplicación, podría ser interesante descubrir, a través del estudio de la red, los grupos que pudieran formarse entre usuarios. Estos grupos podrían deberse a agrupaciones de usuarios en función de similares intereses, y detectar a través de ellos la especialización de perfiles. Por ejemplo, es razonable pensar que alguien interesado en visualización de datos, seguirá o será seguido por usuarios cuyo perfil también incluya un interés en visualización de datos.

Esta parte, sin embargo, requeriría un estudio muy detallado de la estructura de la red, que no es el objetivo principal del proyecto.

6.3 Almacenamiento

La tabla con los usuarios y sus medidas de relevancia se obtiene como un data frame desde el código, y para almacenarlo hemos decidido exportarlo como un fichero Excel, concretamente el fichero **Python/tables/4_ranked_users.xlsx** del repositorio de GitHub, para posteriormente poder importarlo desde R y ofrecer una buena visualización de los datos.

Análogamente, hemos guardado en un fichero el grafo una vez construido a partir de las relaciones entre los usuarios, **Python/graph/relations_graph.gml**. Este fichero se puede importar también desde R y desde Gephy a efectos de visualización.

¹⁰[29], <https://www.sci.unich.it/~francesc/teaching/network/transitivity.html>: por ejemplo, la transitividad de la colaboración entre autores en ciencias de la computación es 0.24, la de la red de colaboración entre actores 0.2 y la de coautores en publicaciones de física, 0.45. Sin embargo, la de Internet es solo 0.012.

6.4 Resultados

De los 155 usuarios que teníamos para ordenar, al intentar bajar desde Twitter los datos (timeline y followers) de alguno de los usuarios, se ha producido un error. En uno de ellos, el texto del error “Not authorized.”, indica que es un usuario con el timeline y resto de datos protegidos. En otro caso, el mensaje de error al acceder a timeline, “[‘code’: 34, ‘message’: ‘Sorry, that page does not exist.’]”, solo parece indicar que es un usuario que no existe más en Twitter. En cualquier caso, son usuarios que no van a aparecer ordenados en nuestra tabla. Mejor dicho, aparecerán, pero sin un valor de ordenación (todos los indicadores a 0).

6.4.1 Ordenación de los usuarios

Para ver la tabla ordenada de los usuarios, lo mejor que puede hacerse es usar el Shiny que hemos desarrollado y que describimos en el siguiente capítulo. Podemos adelantar sin embargo un análisis descriptivo de la tabla de usuarios (por ejemplo, importando la tabla en R):

user_id	h_index	degree	in_degree
Length:155	Min. : 0.0000	Min. : 0.00000	Min. : 0.000000
Class :character	1st Qu.: 0.0000	1st Qu.: 0.00000	1st Qu.: 0.000000
Mode :character	Median : 0.0000	Median : 0.01307	Median : 0.006536
	Mean : 0.1613	Mean : 0.02285	Mean : 0.011427
	3rd Qu.: 0.0000	3rd Qu.: 0.03595	3rd Qu.: 0.019608
	Max. :11.0000	Max. : 0.14379	Max. : 0.071895
out_degree	eigenvector	katz_bonacich	pagerank
Min. : 0.000000	Min. : 0.000000	Min. : 0.00000	Min. : 0.00000
1st Qu.: 0.000000	1st Qu.: 0.000000	1st Qu.: 0.04121	1st Qu.: 0.04972
Median : 0.006536	Median : 0.000000	Median : 0.04699	Median : 0.11092
Mean : 0.011427	Mean : 0.060697	Mean : 0.06558	Mean : 0.18094
3rd Qu.: 0.016340	3rd Qu.: 0.003149	3rd Qu.: 0.07189	3rd Qu.: 0.27053
Max. : 0.071895	Max. : 1.000000	Max. : 0.27634	Max. : 1.00000
closeness	betweenness		
Min. : 0.000000	Min. : 0.000e+00		
1st Qu.: 0.000000	1st Qu.: 0.000e+00		
Median : 0.006536	Median : 0.000e+00		
Mean : 0.027683	Mean : 2.107e-03		
3rd Qu.: 0.069414	3rd Qu.: 7.525e-05		
Max. : 0.116438	Max. : 5.942e-02		

Figura 6.4.1: Análisis descriptivo de los usuarios.

Tal vez lo más llamativo sean los valores del índice h . Para entenderlos un poco mejor, veamos el histograma:

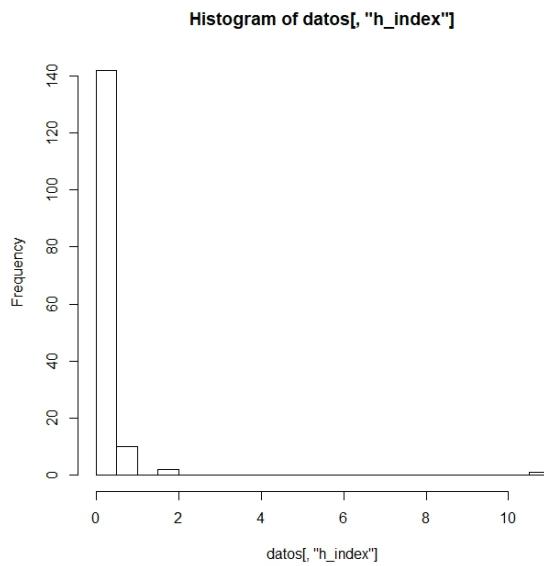


Figura 6.4.2: Histograma de los valores obtenidos del índice h .

A la vista de este gráfico, los valores obtenidos del índice h son nulos en la mayoría de los casos, y solo unos pocos individuos tienen unos valores del índice h distintos de cero. En particular, uno de ellos tiene un valor muy alto, 11. Es el siguiente:



Figura 6.4.3: Usuario con el índice h más alto.

Con respecto al resto de las medidas, hemos procurado que el código nos devuelva valores normalizados de las centralidades. Nuestra idea es no sólo ordenar los usuarios por sus niveles individuales de cada uno de los índices, sino también poder construir índices personalizados, asignando pesos distintos a cada uno de los índices, y luego combinándolos. Para ello, el siguiente paso, que se lleva a cabo en la última fase, de visualización, es la normalización también de los niveles del índice h .

6.4.2 Propiedades del grafo

El grafo que hemos obtenido está formado por 154 nodos (todos los usuarios, menos aquel cuyos datos están protegidos en Twitter) y 271 arcos.

El grafo no es fuertemente conexo, y tiene 91 componentes fuertemente conexas. La mayor de las componentes tiene 35 nodos, que son el 22.73% del total. Se trata de un grafo bastante desconectado. No es ninguna sorpresa, ya que estamos buscando las conexiones entre los usuarios que hemos seleccionado a través de las fases que hemos detallado a lo largo de esta memoria.

Para la componente conexa de mayor tamaño, las medidas relevantes son las siguientes (recordar que en la sección [6.2.2](#) hemos revisado su significado):

- radio: 3
- diámetro: 6
- longitud media del camino más corto: 3

Dado que estamos en un grafo dirigido, podemos preguntarnos por la conectividad débil del mismo. El grafo que hemos obtenido no es tampoco débilmente conexo, y cuenta con 61 componentes débilmente conexas. La mayor de ellas agrupa a 76 nodos, algo más del 49% del total.

Finalmente, el coeficiente de transitividad del grafo es 0.497.

A continuación, mostramos los histogramas con las distribuciones de los grados de los nodos y las longitudes de los caminos mínimos

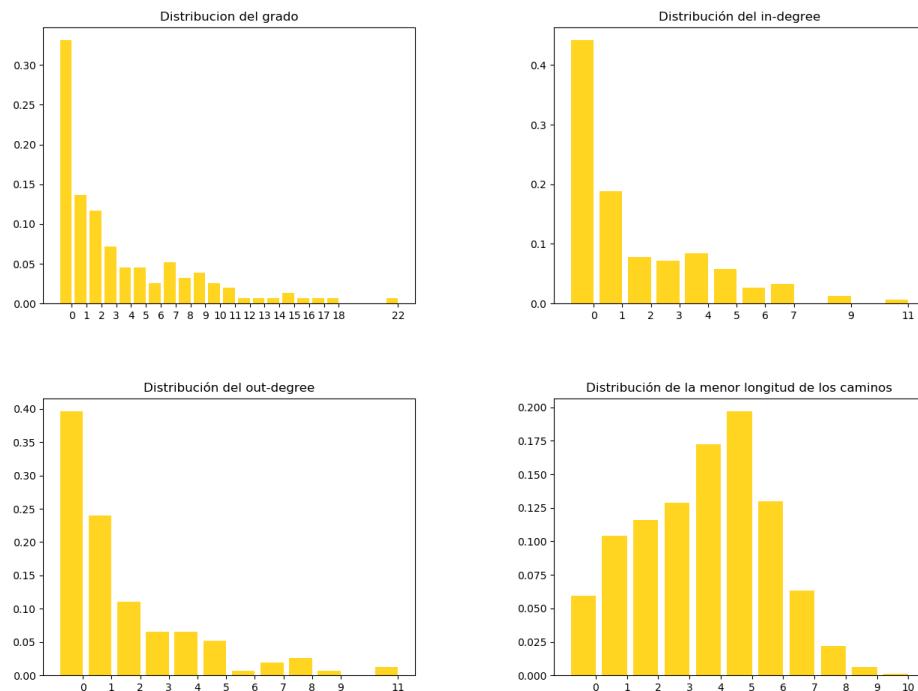


Figura 6.4.4: Distribuciones de varias medidas de los nodos del grafo.

Capítulo 7

Visualización de los resultados

7.1 Herramientas

Para visualizar la tabla de los usuarios ordenados a través de su índice h y de sus centralidades hemos recurrido al paquete Shiny de R. El código con el desarrollo de esta visualización está en la carpeta **Visualizacion/Shiny** de nuestro repositorio de GitHub.

7.2 Descripción de la interfaz

La información a mostrar al potencial cliente del proyecto es por un lado la lista de usuarios de Twitter susceptibles de ser candidatos a la oferta de trabajo y por otro, de forma accesoria, el grafo que muestra la relación entre la comunidad de usuarios identificados a lo largo del proceso de selección de dichos usuarios. La interfaz dispone por ello de dos pestañas, “Listado de candidatos” y “Grafo”:

N	Id	User_id	h_index	Degree	In_degree	Out_degree	Eigenvector	Katz_bonacich	Pagerank	Closeness	Beta
1	0	100280852	0	0	0	0	0	0.041	0.05	0	
2	1	11025592	0	0.052	0.026	0.026	0.022	0.076	0.593	0.094	
3	2	1110307802	0	0	0	0	0	0.041	0.05	0	
4	3	111922242	1	0.072	0.026	0.046	0.023	0.08	0.271	0.099	
5	4	115027611	0	0	0	0	0	0.041	0.05	0	
6	5	119260968	1	0	0	0	0	0.041	0.05	0	
7	6	119897328	0	0.02	0.007	0.013	0	0.047	0.144	0.007	

Figura 7.2.1: Aspecto general de la visualización, con las dos pestañas disponibles

Antes de describir estas dos pestañas en detalle, queremos hacer hincapié en que la información relativa a los usuarios siempre se da en relación a su “user.id”. En ningún caso proporcionamos nombres ni información obtenida personal directamente del análisis. En caso de querer contactar con ese usuario se tendría que hacer a través del propio Twitter y una vez obtenido el consentimiento del usuario a través de cualquier otra vía disponible (LinkedIn, blog, e-mail...)¹.

7.2.1 La pestaña con la lista de usuarios ordenados: “Listado de candidatos”

En esta pestaña aparece una tabla con diversos campos, comenzando por el campo “user.id” que identifica al usuario, y al que siguen varias columnas con el índice h y los diferentes tipos de centralidades. Las definiciones de las cantidades reflejadas en cada columna se pueden obtener pasando el ratón por encima de los títulos de la tabla para que el cliente pueda tener a mano su significado sin necesidad de ir a otro sitio:

Ranking de candidatos											
Listado de candidatos			Grafo								
Pondera el resultado											
Show 10 entries									Search:		
N	Id	User_id	h_index	Degree	In_degree	Out_degree	Eigenvector	Katz_bonacich	PageRank		
1	0	100280852	Sistema propuesto para la medición de la calidad profesional de físicos y de otros científicos en función de la cantidad de citas que han recibido sus artículos científicos. Un científico tiene índice h si ha publicado h trabajos con al menos h citas cada uno.								
2	1	11025592	26								
3	2	1110307802	0								
4	3	111922242	1	0.072	0.026	0.046	0.023	0.08	0.271		
5	4	115027611	0	0	0	0	0	0.041	0.05		
6	5	119260968	1	0	0	0	0	0.041	0.05		
7	6	119897328	0	0.02	0.007	0.013	0	0.047	0.144		

Figura 7.2.2: La definición de cada campo se puede ver al pasar el puntero del ratón por encima del título de la columna.

La pestaña permite la selección del número de usuarios a mostrar por página: 10, 25, 50 o 100 usuarios por página:

¹ Esta decisión la explicamos con detalle en la página 20

Ranking de candidatos

Listado de candidatos		Grafo	Pondera el resultado						
Show	10 entries								
N	25	User_id	h_index	Degree	In_degree	Out_degree	Eigenvector	Katz_bonacich	Page
1	1	100280852	0	0	0	0	0	0	0.041
2	1	11025592	0	0.052	0.026	0.026	0.022	0.076	
3	2	1110307802	0	0	0	0	0	0	0.041
4	3	111922242	1	0.072	0.026	0.046	0.023	0.08	
5	4	115027611	0	0	0	0	0	0	0.041
6	5	119260968	1	0	0	0	0	0	0.041
7	6	119897328	0	0.02	0.007	0.013	0	0	0.047

Figura 7.2.3: Elección del número de registros por página a mostrar.

La aplicación también permite buscar por cualquier valor mostrado:

Ranking de candidatos

Listado de candidatos		Grafo	Pondera el resultado								
Show	10 entries										
			Search: 0.41								
N	Id	User_id	h_index	Degree	In_degree	Out_degree	Eigenvector	Katz_bonacich	PageRank	Closeness	Bet
35	34	207676479	0	0.059	0.059	0	0.582	0.192	0.412	0.095	
69	68	287285569	0	0.039	0.026	0.013	0.412	0.138	0.193	0.087	

Showing 1 to 2 of 2 entries (filtered from 154 total entries) Previous 1 Next

Figura 7.2.4: Búsqueda por valores mostrados.

Para ordenar a los usuarios por los valores de alguna de las columnas, basta con usar las flechas que hay al lado de los nombres de las columnas (también se puede ordenar por varias columnas y con diferentes criterios, orden ascendente o descendente):

Show 10 entries

N	Id	User_id	h_index	Degree
1	0	100280852	0	0
2	1	11025592	0	0.052

Figura 7.2.5: Flechas para ordenar los usuarios por una única columna.

Además de ese orden sencillo hemos incluido una columna “resultado” donde ponderamos el resultado en función de los distintos índices que el cliente escoja como más relevantes para la selección adecuada del candidato. Para poder calcularlo adecuadamente hemos normalizado los datos ya que las escalas entre el índice h y las centralidades era muy distintos. Hemos incluido también varios avisos para avisar al usuario en el caso de que los datos introducidos no sean correctos. Para acceder a la pantalla donde poder introducir los distintos pesos, hay que hacer click en el botón “Pondera el resultado”, y aparece la siguiente pantalla:

Ranking de candidatos

Listado de candidatos Grafo

Pondera el resultado

Show 10 entries

N	Id	User_id	h_index	Degree	In_degree	Out_degree	Eigenvector
1	0	100280852	0	0	0	0	0
2	1	11025592	0	0.052	0.026	0.026	0.022
3	2	1110307802	0	0	0	0	0

Figura 7.2.6: Acceso a la pantalla para introducir los pesos.

En la ventana emergente se indica que los datos a introducir deben sumar 100. Si por error no es así, vuelve a aparecer otra ventana adicional que indica que los resultados se deben revisar para que sumen 100.

Rank

Introduce porcentajes para ponderar el resultado

La suma de los valores introducidos debe ser 100

h_index	Degree	In_degree
0	0	0
Out_degree	Eigenvector	Katz_Bonacich
0	0	0
Out_degree	PageRank	Closeness
0	0	0.094
Betweeness		0.095
0		0.007
Betweeness		0.021
0		0.018

Pulsa ‘Calcular’ y cierra la ventana para ver el resultado

Calcular

Figura 7.2.7: Pantalla para introducir los pesos.

Si a pesar de ello no hacemos caso de estas advertencias en la página principal aparece un mensaje en rojo donde se indica claramente que la suma de los porcentajes no es correcta (aunque el resultado de la ponderación, sume 100 o no, se mostrará siempre en la columna "Resultado"):

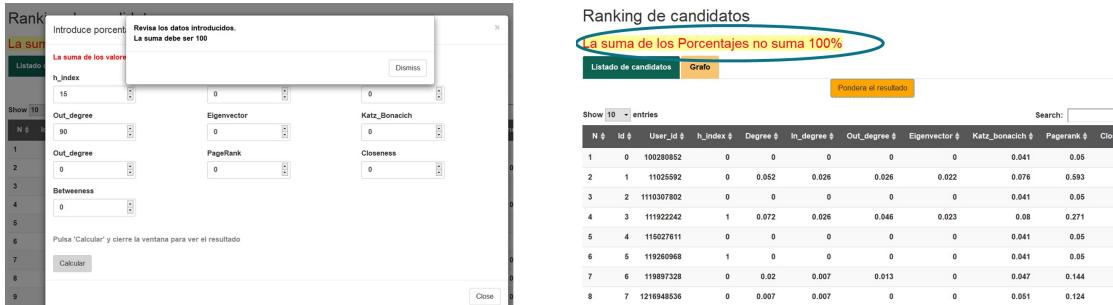


Figura 7.2.8: Mensajes de error en el caso de que los pesos no sean valores válidos.

7.2.2 La estructura de la red: pestaña “Grafo”

En la pestaña “Grafo” aparece una representación visual de los candidatos y sus relaciones:

Ranking de candidatos



Figura 7.2.9: Aspecto general de la pantalla donde se muestra el grafo.

Aunque el grafo que construimos con las relaciones entre los usuarios es un grafo dirigido (tiene distinta importancia si un usuario sigue o es seguido), la representación gráfica de ese grafo, con flechas, es bastante farragosa. Para mejorar la calidad gráfica, hemos elegido representar el grafo como un grafo no dirigido.

Los nodos están representados con diferentes colores y tamaños, que son función del índice h . Pasando el puntero del ratón por encima de cada nodo aparece su “user.id”:

Ranking de candidatos

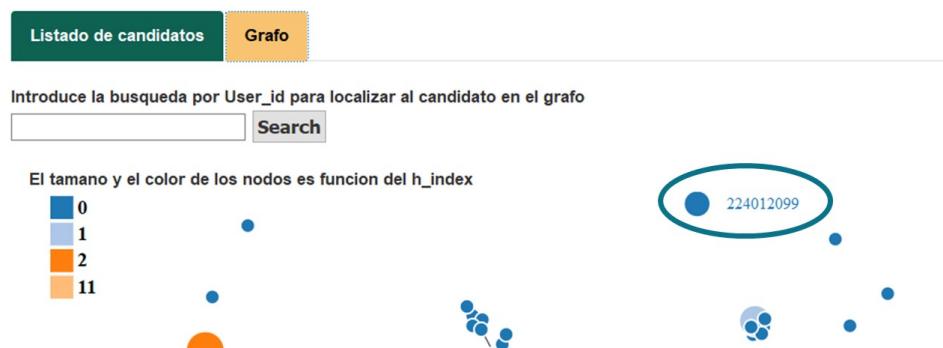


Figura 7.2.10: “user.id” de un nodo por el que hemos pasado el puntero.

Y haciendo doble click sobre un nodo, aparece toda su información (la que tenemos en la tabla):

Ranking de candidatos

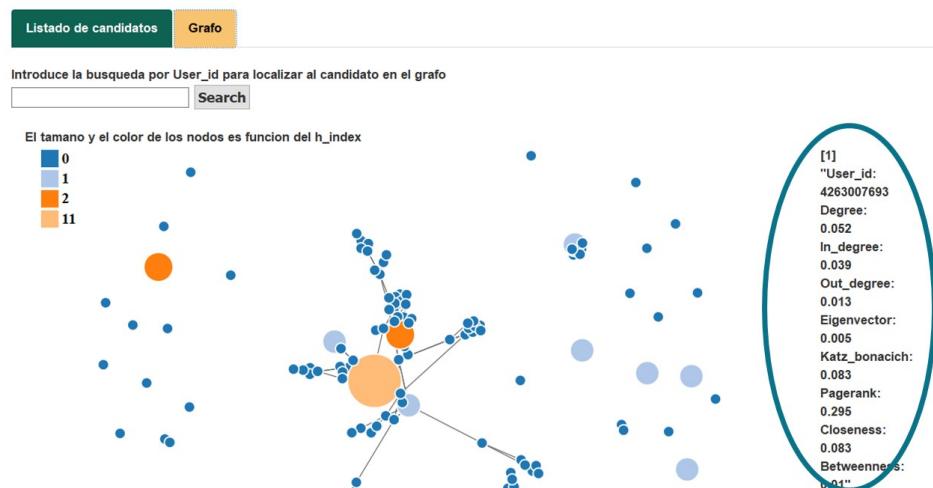


Figura 7.2.11: Información de un nodo en el que hemos hecho doble click.

Si queremos localizar a un nodo por su “user.id”, podemos introducirlo en la casilla de búsqueda y al pulsar “Search”, desaparecerán todos los nodos quedándose tan solo resaltado el nodo buscado. Esto durará unos segundos hasta que el grafo vuelve a aparecer normalmente:

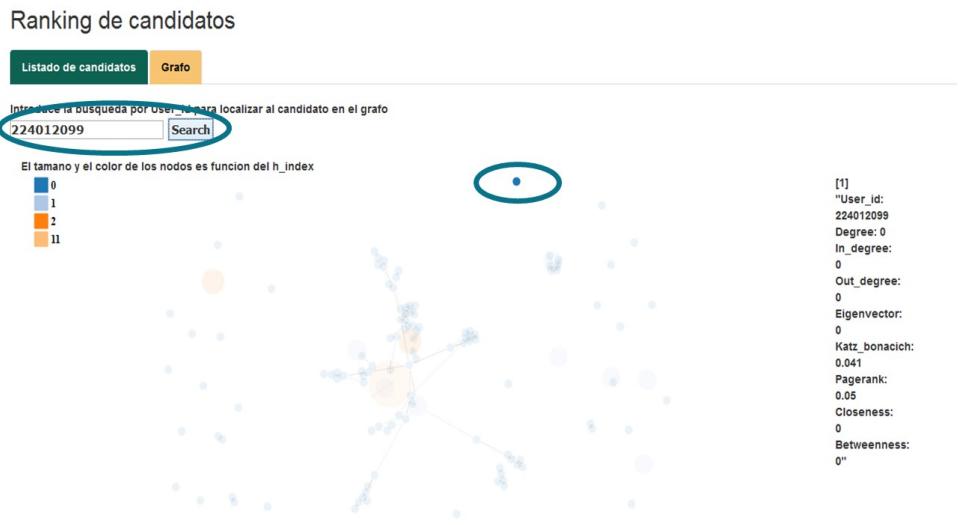


Figura 7.2.12: Búsqueda de un usuario en el grafo por su “user.id”.

Del grafo cabe destacar los siguientes comentarios:

- Pocos nodos (8%) tienen un índice h igual o por encima de 1. Teniendo en cuenta la definición del índice h (sección 6.1), eso significa que muchos de los usuarios seleccionados no tienen retuits de sus publicaciones.
- El único usuario con un índice h notable (11) tiene un in-degree de 0.039 (no demasiado alto). Es decir, que este usuario no es de los que más seguidores tiene entre los usuarios destacados. Sin embargo de sus publicaciones sobre el tema de referencia, 11 de ellas han sido retuiteadas al menos 11 veces. Además, mirando su intermediación (betweenness) tampoco es de las más altas, con lo que no conecta muchos de los otros usuarios entre sí. Este usuario, cuya relevancia es notable en la comunidad general (más allá de la incluida en nuestra muestra), se captó a través de otro el cual le citaba en sus tuits.
- El usuario con mayor in-degree (mayor número de seguidores de entre los seleccionados) coincide con el de mayor out-degree (mayor número de amigos de entre los seleccionados). Se trata de la directora de una empresa del sector.
- El usuario con más intermediación (influencia para compartir información entre el resto de usuarios) se trata de un profesional del marketing y BI que dispone de su propio blog.
- Varios de los usuarios con índice $h=1$ y uno con índice $h=2$ estás “desconectados” de la red o casi “desconectados”. Esto puede ser porque el tiempo en el que duró la descarga de datos de twitter no captó al resto de sus contactos.
- Por último mencionar que los valores tanto de “pagerank” como de “katz_bonacich” nunca serán igual a cero, aunque el in-degree y el out-degree del nodo sean cero, ya que asignan una centralidad mínima.

Para una aplicación con un número elevado de usuarios y relaciones, el grafo solo se mostrará para una cantidad de usuarios que aporten mayor relevancia en cualquiera de los índices.

Capítulo 8

Áreas de mejora

El proyecto desarrollado es meramente un prototipo de una aplicación. En casi todas las fases del proyecto hay aspectos que podrían mejorarse. A continuación, exponemos algunas de ellas:

- Por supuesto, implementar los recursos adecuados para que la LOPD no sea un obstáculo para la comercialización del proyecto.
- Usar un acceso a Twitter que nos permita descargar de forma exhaustiva la información en relación al perfil de referencia de la oferta de trabajo.
- Implementar una estructura más escalable, en la nube, para el manejo de un mayor volumen de datos.
- En la fase de selección de usuarios:
 - Detección del lenguaje: usar corpus etiquetados para entrenar un modelo, o usar uno más adecuado para el tipo de lenguaje de los usuarios de Twitter (tipo equilid).
 - Almacenamiento: usar una base de datos SQL para almacenar los resultados intermedios en el proceso, en la nube para mejorar la escalabilidad.
 - Tipo de usuario: explorar otras formas de detectar personas, empresas, bots, etc.. Sería interesante incluir un código de reconocimiento facial para identificar si en la foto del perfil aparece una persona y añadir el resultado al modelo. También refinar los criterios del modelo y explorar nuevos criterios.
 - Naturaleza del tuit: mejorar el modelo para conseguir mayor granularidad en la clasificación y detectar distintas especialidades dentro del perfil de referencia.
- En la fase de ordenación de usuarios: explotar más profundamente la información del grafo, añadiendo características a los nodos, y usar técnicas de detección de comunidades también para detectar especialidades.
- En la fase de visualización, adaptar la visualización a un entorno productivo, quizá incluyendo visualización a través de una página web.

- Extender la funcionalidad incluida en el proyecto: quizá una monitorización continua de las publicaciones en Twitter sobre contenidos relacionados con los perfiles de referencia para detectar nuevos usuarios relacionados, e ir aumentando el universo de los potenciales candidatos (para un servicio continuado a clientes).
- Añadir otras características a la clasificación final de los candidatos: por ejemplo, investigar a partir del nombre de usuario de Twitter si dicho usuario tiene un repositorio en Github o un usuario activo en StackOverflow para reforzar la relevancia de su perfil.

8.1 Mejoras en el algoritmo de clasificación del contenido del tuit

En esta sección explicaremos cómo hemos comenzado a trabajar en la mejora del modelo de clasificación descrito en la sección 5.2.1. Para ello, hemos hecho pruebas con el mismo modelo usado en esa sección, el Multinomial NB, y con otro modelo de clasificación, un SVM, tratando de mejorar los resultados de dos maneras:

- buscando los mejores parámetros para optimizar la eficiencia del modelo (Hyperparameters Optimization) a través del método `GridSearchCV()` y
- cambiando el método de entrenamiento, añadiendo al método sencillo de separación entre entrenamiento y test (que habíamos establecido en 70/30), un algoritmo “*K-fold cross validation*”, que separa el conjunto de datos en K “folds” o subconjuntos de igual tamaño y cada “fold” actúa una vez como conjunto de test y $K - 1$ veces como parte del conjunto de entrenamiento.

Mejorando el modelo `MultinomialNB()` con `GridSearchCV()` y “*K-fold cross validation*”

Con el siguiente código especificamos los parámetros a optimizar, y el método para hacerlo, un método “*K-fold cross validation*” (en concreto, con la opción “`cv = 5`”):

```
parameters = {'tfidf_ngram_range': [(1,1),(1,2)],
              'tfidf_use_idf': (True, False),
              'tfidf_sublinear_tf': (True, False),
              'tfidf_min_df':(0,5,10),
              'tfidf_max_df':(0.95,0.90,0.85),
              'clf_alpha': (0.001,0.01,0.1,1,10),
              }
text_cl = Pipeline([('tfidf', TfidfVectorizer()),('clf', MultinomialNB())])
gs_clf = GridSearchCV(text_cl, parameters, cv = 5, scoring='roc_auc', n_jobs=-1)
gs_clf = gs_clf.fit(x_train, y_train)
# Aplicando el modelo en la base de test
gs_pred = gs_clf.predict(x_test)
```

La última línea es la aplicación del modelo al conjunto de test (el 30% de los tuits), y obtenemos estos resultados:

Matriz de confusión		Métricas de clasificación				
		Precisión	Recall	F1-score	Support	
	1 0	0	0.95	0.99	0.97	528
1	524 4	1	0.91	0.58	0.71	72
0	30 42	Avg/total	0.94	0.94	0.94	600

Respecto a la curva ROC, hemos obtenido una exactitud ("accuracy") de 0.910208859428, y la siguiente curva ROC:

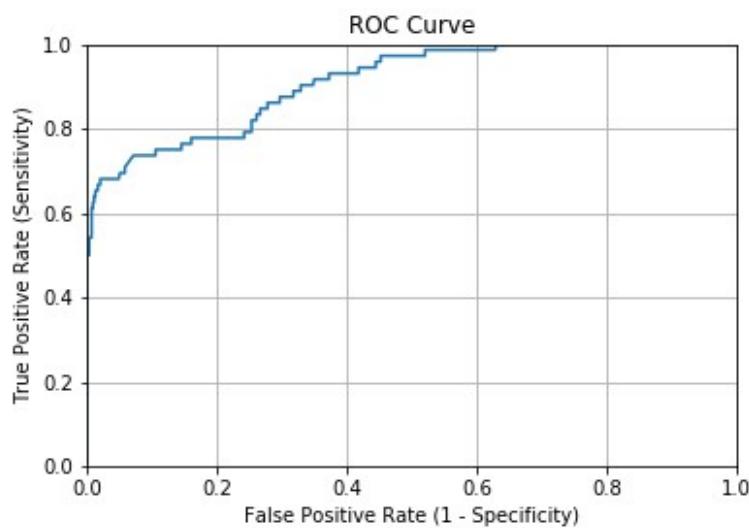


Figura 8.1.1: Curva ROC de la primera mejora del modelo de clasificación de textos según su relevancia, MultinomialNB().

Cambiamos el modelo a SVM con GridSearchCV(), también con “K-fold cross validation”

El código para entrenar este modelo es muy similar al del ejemplo anterior, y de nuevo tenemos unos parámetros que indican cuáles vamos a optimizar, y una opción ("cv=5") para incluir el algoritmo del "K-fold cross validation":

```
parameters = {'tfidf_ngram_range': [(1,1),(1,2)],
              'tfidf_use_idf': (True, False),
              'tfidf_sublinear_tf': (True, False),
              'tfidf_min_df':(0,5,10),
              'tfidf_max_df':(0.95,0.90,0.85),
              'clf_C': (1,10,100,1000),
              'clf_tol':(0.001,0.01,0.1),
              }
```

```

text_cl = Pipeline([('tfidf', TfidfVectorizer()), ('clf', LinearSVC())])
gs_clf = GridSearchCV(text_cl, parameters, cv = 5, scoring='roc_auc', n_jobs=-1)
gs_clf = gs_clf.fit(x_train, y_train)
# Aplicando el modelo en la base de test
gs_pred = gs_clf.predict(x_test)

```

A partir de la última línea, la aplicación del modelo al conjunto de test, obtenemos estos resultados:

Matriz de confusión		Métricas de clasificación			
		Precisión	Recall	F1-score	Support
	0	0.95	0.99	0.97	528
1	521	0.87	0.64	0.74	72
0	26	0.94	0.94	0.94	600
Avg/total		0.94	0.94	0.94	600

En la curva ROC para este modelo, hemos obtenido una exactitud (“accuracy”) de 0.952033354377, y la siguiente gráfica:

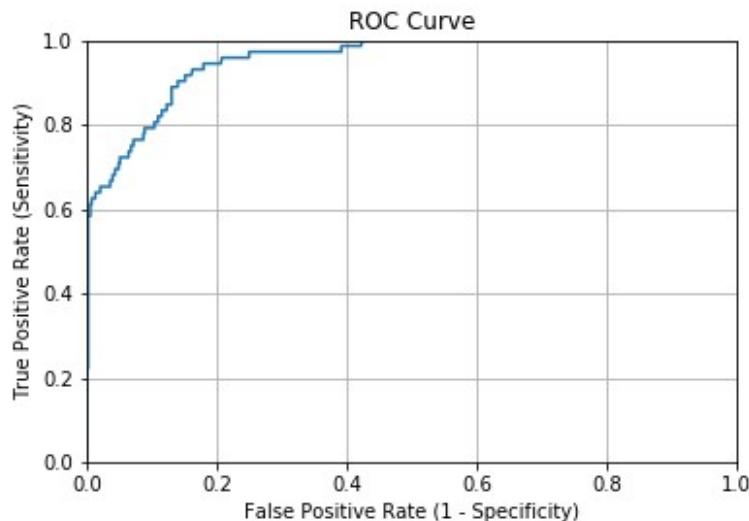


Figura 8.1.2: Curva ROC del modelo LinearSVC().

Conclusión

Los dos modelos han resultado producir una “accuracy” y métricas muy parecidas. El modelo SVM tal vez aventaje ligeramente al Naïve Bayes. Sin embargo, el Naïve Bayes tiene un procesamiento más rápido y sencillo. Por eso creemos que será necesario incluir los dos modelos en el proyecto para llegar a una conclusión, pero si en el futuro es necesario tener velocidad, usaríamos el Naïve Bayes.

Bibliografía

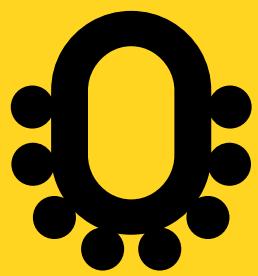
- [1] María Gloria Castaño collado, Gerardo de la Merced López Montalvo, José María Prieto Zamora, *Guía técnica y de buenas prácticas en reclutamiento y selección de personal (R& S)*. Documento aprobado por la Junta de Gobierno del Colegio Oficial de Psicólogos de Madrid, Febrero de 2011. <http://www.copmadrid.org/webcopm/recursos/guiatecnicabuenaspracticas.pdf>
- [2] *Selección de personal para no especialistas*. Andalucía Emprende, Fundación Pública Andaluza. Consejería de Economía y Conocimiento. <https://www.andaluciaemprende.es/wp-content/uploads/2015/02/guia\discretionary{-}{}{}{}seleccion-personal.pdf>
- [3] *Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal*. Jefatura del Estado BOE núm. 298, de 14 de diciembre de 1999 Referencia: BOE-A-1999-23750 http://www.agpd.es/portalwebAGPD/canaldocumentacion/legislacion/estatal/common/pdfs/2014/Ley_Organica_15-1999_de_13_de_diciembre_de_Proteccion_de_Datos_Consolidado.pdf
- [4] María Luz Congosto Martínez, *Caracterización de usuarios y propagación de mensajes en Twitter en el entorno de temas sociales*. Tesis doctoral.
- [5] “Twitter”. Wikipedia. <https://es.wikipedia.org/wiki/Twitter>.
- [6] Shamanth Kumar, Fred Morstatter, Huan Liu. *Twitter Data Analytics*. Springer (2013).
- [7] Twitter Developer Documentation<https://dev.twitter.com/>
- [8] Steven Bird, Ewan Klein, Edward Loper. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly (2009). <http://www.nltk.org/book/>
- [9] Marc A. Zissman, Kay M. Berkling. Automatic language identification. *Speech Communication*, Volume 35, Issues 1–2, August 2001, Pg. 115-124.
- [10] Y. Almeida-Cruz, S. Estévez-Velarde, A. Piad-Morffis. Detección de Idioma en Twitter. *GECONTEC: Revista Internacional de Gestión del Conocimiento y la Tecnología*, Vol.2 (3), 2014. https://www.upo.es/revistas/index.php/gecontec/article/view/1081/pdf_11

- [11] Marco Lui, Timothy Baldwin. langid.py: An Off-the-shelf Language Identification Tool. *Proceedings of the ACL 2012 System Demonstrations*, pg. 25–30, 2012. <http://www.aclweb.org/anthology/P12-3005>
- [12] Marco Lui, Timothy Baldwin. Accurate Language Identification of Twitter Messages. *Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM) @ EACL 2014*, pg. 17–25, (2014). <http://www.aclweb.org/anthology/W14-1303>
- [13] David Jurgens, Yulia Tsvetkov, Dan Jurafsky. Incorporating Dialectal Variability for Socially Equitable Language Identification. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Short Papers)*, pg. 51–57, (2017). <https://doi.org/10.18653/v1/P17-2009>
- [14] Tobias M. Scholz. *Big Data in Organizations and the Role of Human Resource Management*, Peter Lang Academic Research, Series: Personalmanagement und Organisation, Vol. 5 (2017).
- [15] Marco Pennacchiotti, Ana-Maria Popescu. A Machine Learning Approach to Twitter User Classification, AAAI Publications, Fifth International AAAI Conference on Weblogs and Social Media (2011). <https://webpages.uncc.edu/anraja/courses/SMS/SMSBib/2886-14198-1-PB.pdf>
- [16] Anjie Fang, Iadh Ounis, Philip Habel, Craig Macdonald, Nut Limsopatham. Topic-centric Classification of Twitter User's Political Orientation. *Proceedings of the 6th Symposium on Future Directions in Information Access* (2015). <http://www.dcs.gla.ac.uk/~anjie/papers/fang2015fdia.pdf>
- [17] Tomoya Noro, Atsushi Mizuoka, Takehiro Tokuda. Towards Finding Good Twitter Users to Follow Based on User Classification. *Proceedings of The 24th International Conference on Information Modelling and Knowledge Bases* (2014). https://www.researchgate.net/publication/269994032_Towards_Finding_Good_Twitter_Users_to_Follow_Based_on_User_Classification
- [18] Zi Chu, Steven Gianvecchio, Haining Wang, Sushil Jajodia. Detecting Automation of Twitter Accounts: Are You a Human, Bot, or Cyborg?. IEEE Transactions on Dependable and Secure Computing, Vol. 9 (2012). <http://www.cs.wm.edu/~hnw/paper/tdsc12b.pdf>.
- [19] Yan L., Ma Q., Yoshikawa M.. Classifying Twitter Users Based on User Profile and Followers Distribution. En: Decker H., Lhotská L., Link S., Basl J., Tjoa A.M. (eds) Database and Expert Systems Applications. DEXA 2013. Lecture Notes in Computer Science, vol 8055. Springer (2013). https://link.springer.com/chapter/10.1007/978-3-642-40285-2_34.
- [20] Zafar Gilani, Ekaterina Kochmar, Jon Crowcroft. Classification of Twitter Accounts into Automated Agents and Human Users. *Proceedings of the 9th IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM'17 (2017). https://www.cl.cam.ac.uk/~szuhg2/docs/papers/ASONAM17_8501_66_1.pdf

- [21] Munmun De Choudhury, Nicholas Diakopoulos, Mor Naaman. Unfolding the Event Landscape on Twitter: Classification and Exploration of User Categories. *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, pg. 241-244 (2012). http://www.munmund.net/pubs/cscw_2012.pdf#view=Fit
- [22] Raviv Cohen, Derek Ruths. Classifying Political Orientation on Twitter: It's Not Easy!. *Proceedings of the Seventh International AAAI Conference on Weblogs and Social Media* (2013). <https://www.aaai.org/ocs/index.php/ICWSM/ICWSM13/paper/viewFile/6128/6347>
- [23] Morgane Ciot, Morgan Sonderegger, Derek Ruths. Gender Inference of Twitter Users in Non-English Contexts. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pg. 1136–1145 (2013). <http://www.aclweb.org/anthology/D13-1114>
- [24] Wendy Liu, Derek Ruths. What's in a Name? Using First Names as Features for Gender Inference in Twitter. *Analyzing Microtext: Papers from the 2013 AAAI Spring Symposium* (2013). <https://www.aaai.org/ocs/index.php/SSS/SSS13/paper/view/5744/5908>
- [25] Clay Fink, Jonathon Kopecky, Maksym Morawski. Inferring Gender from the Content of Tweets: A Region Specific Example. *Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media* (2012). <http://www.aaai.org/ocs/index.php/ICWSM/ICWSM12/paper/download/4644/5031>
- [26] Shaomei Wu, Jake M. Hofman, Winter A. Mason, Duncan J. Watts. Who Says What to Whom on Twitter. *Proceedings of the 20th International Conference on World Wide Web* (2011). <http://www.wwwconference.org/proceedings/www2011/proceedings/p705.pdf>
- [27] Carlos A. Freitas, Fabrício Benevenuto, Saptarshi Ghosh, Adriano Veloso. Reverse Engineering Socialbot Infiltration Strategies in Twitter. *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pg. 25-32 (2015). https://socialnetworks.mpi-sws.org/papers/TwitterBots_ASONAM15.pdf
- [28] Jasmit Kaur, Alexis A. Fink. Trends and Practices in Talent Analytics. SHRM-SIOP Science of HR White Paper Series (2017). http://www.siop.org/SIOP-SHRM/2017%202010_SHRM-SIOP%20Talent%20Analytics.pdf
- [29] Fernando Pérez García. Teoría de Grafos. Notas de las clases impartidas en el Máster en Data Science, MBIT, convocatoria Marzo 2017.
- [30] Phillip Bonacich, Paulette Lloyd. Eigenvector-like measures of centrality for asymmetric relations. *Social Networks*, 23, pg. 191–201 (2001). <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.226.2113&rep=rep1&type=pdf>
- [31] Phillip Bonacich. A Family of Measures. *American Journal of Sociology*, Vol. 92, No. 5, pg 1170-1182 (1987). <http://www.leonidzhukov.net/hse/2014/socialnetworks/papers/Bonacich-Centrality.pdf>.

- [32] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, Vol. 46, Issue 5 (1999). <http://www.cs.ucr.edu/~vagelis/classes/CS172/publications/kleinberg98authoritative.pdf>
- [33] Lalindra De Silva, Ellen Riloff. User Type Classification of Tweets with Implications for Event Recognition. Proceedings of the Joint Workshop on Social Dynamics and Personal Attributes in Social Media (2014). <https://www.cs.utah.edu/~riloff/pdfs/DeSilva-SMWorkshop-ACL14.pdf>
- [34] Pablo Fernández Gallardo. Google's secret and Linear Algebra, EMS Newsletter, vol. 63, pg. 10-15 (2007)
- [35] Álvaro Romero. Sistemas de Recomendación. Notas de las clases impartidas en el Máster en Data Science, MBIT, convocatoria Marzo 2017.
- [36] Antonio LaTorre. Aprendizaje supervisado. Notas de las clases impartidas en el Máster en Data Science, MBIT, convocatoria Marzo 2017.

Documento producido con L^AT_EX.



OCTOPUS
DATA INSIGHTS