Algoritmos de clustering y aprendizaje automático aplicados a Twitter

Eric-Joel Blanco-Hermida Sanz

Enero 2016

UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Agradecimientos

Quiero agradecer a mis padres que siempre me han apoyado en todo.

A Kevin Desir con el que compartimos mucho en el tiempo que estuvimos en la empresa.

A toda la gente de Sogeti High Tech que me hicieron sentir muy a gusto.

Por ultimo agradecer a Marta la ayuda y los consejos para sacar adelante este proyecto.

Resumen

En este proyecto llevado a cabo en la empresa Sogeti High Tech en colaboración con Orange, hemos realizado un estudio sobre las posibilidades de algoritmos de clustering y aprendizaje automático aplicadas a la red social Twitter. Primero hemos hecho un estado del arte de los algoritmos más importantes de clustering y de aprendizaje automático así como evaluar los frameworks de machine learning más populares. Hemos recolectado millones de tweets que contuviesen la palabra "Orange" para después , mediante algoritmos de aprendizaje supervisado, distinguir aquellos que se refieren a la empresa de los que no. También hemos hecho análisis de sentimiento de los tweets a fin de ver cuando un tweet habla de forma positiva o negativa de la empresa. Finalmente, hemos aplicado clustering a los tweets para ver que información podíamos extraer de los datos.

Abstract

In this project made at Sogeti High Tech in collaboration with Orange, we have made a study on clustering and machine learning algorithms applied to Twitter. Firstly we made a state of the art of the most important clustering and machine learning algorithms and a benchmark of the most popular machine learning frameworks. We collected millions of tweets that contained the word "Orange" for, using supervised learning algorithms, be able to tell apart the ones that refer to the enterprise that the ones who don't. We also did sentiment analysis on the tweets to see whether they are talking positively or negatively about the enterprise. Finally we applied clustering to those tweets in order to see which kind of information we can obtain from data.

Resum

En aquest projecte fet a la empresa Sogeti High Tech en col·laboració amb Orange, hem realitzat un estudi sobre las possibilitats dels algoritmes de clustering y aprenentatge automàtic aplicats a la xarxa social Twitter. Primerament hem fet un estat de l' art dels algoritmes mes importants de clustering i de aprenentatge automàtic aixi com una avaluació dels frameworks de machine learning mes populars. Hem recol·lectat milions de tweets que contenien la paraula "Orange" per després , mitjançant algoritmes d'aprenentatge

supervisat, fer la distinció entre aquells que parlen de l'empresa dels que no. També hem fet anàlisis de sentiment dels tweets per veure si es parla de forma positiva o negativa de l'empresa. Finalment, hem aplicat clustering als tweets per veure quina informació podíem extraure.

Contents

1 Introducción

Con el rápido crecimiento de información en Internet[1], la minería de datos[2] se ha convertido en un campo muy importante para aprovechar esa gran cantidad de datos. Nos permite descubrir nuevos patrones o extraer información relevante sobre los datos de forma automática.

El aprendizaje automático es una serie de técnicas para que una maquina pueda aprender y generalizar comportamientos a partir de información suministrada. Los algoritmos aprenden de los datos para poder hacer predicciones. Dentro del aprendizaje automático nos encontramos distintos métodos. Uno de ellos es el aprendizaje supervisado.

El aprendizaje supervisado[3] usa técnicas para deducir una función a partir de datos de entrenamiento. El objetivo es predecir el valor correspondiente a cualquier objeto de entrada a partir de los datos de entrenamiento que ha visto previamente. Un ejemplo de este tipo de algoritmo es el problema de la clasificación, donde el sistema de aprendizaje etiqueta los datos de entrada basándose en los ejemplos ya etiquetados que le hemos suministrado para que aprenda.

Otro método importante del aprendizaje automático es el aprendizaje no supervisado[4]. En contraposición al anterior método, éste no tiene una base de conocimiento previo, no le suministramos datos de entrenamiento. Los datos no tienen etiqueta y por tanto es difícil evaluar la calidad de una solución al no poder saber si ha cometido un error al clasificar.

La empresa Sogeti High Tech[1] me ha propuesto realizar un proyecto de investigación de algoritmos de clustering y de aprendizaje automático. Este proyecto se llevará a cabo en colaboración con la empresa Orange[2].

1.1 Formulación del problema

La empresa francesa Orange tiene un motor de búsqueda (lemoteur.fr)[5]. Este motor data de 1996 con lo que es anterior a Google. Actualmente es el

[1] http://www.sogeti-hightech.fr/en/
[2] http://www.orange.fr/

4º motor de búsqueda mas utilizado en Francia[6].

Inicialmente debía realizar un proyecto relacionado con el motor de búsqueda pero debido ciertas circunstancias el proyecto ha cambiado completamente.

El proyecto que me han encomendado es ver las posibilidades que ofrecen los algoritmos de clustering y aprendizaje automático aplicados a la red social Twitter. En concreto, hacer clustering sobre lo que se dice en Twitter de Orange. Además de esto, hay que realizar análisis de sentimiento de los tweets mediante aprendizaje automático.

Para ello hay que utilizar alguno de los muchos frameworks de aprendizaje automático que son libres que hay en el mercado. Los más notables son Weka[3] y Scikit-learn[4].

Hay que por tanto recuperar tweets que hablen de la empresa y aplicar algoritmos de clustering o de aprendizaje supervisado para resolver esta tarea. En este sentido, los métodos no cambian, cambia el objetivo final por lo que aprovechamos el trabajo realizado completamente.

1.2 Actores

La siguiente es una lista de los actores implicados en el proyecto.

Desarrolladores: Las funciones de desarrollador, diseñador y tester serán realizadas por mi, dado que soy el único asignado al proyecto.

Jefe de proyecto: El jefe de proyecto es Jean-Luc Mouetaux, es el quien se encarga de hacer reuniones de seguimiento, de marcar los objetivos y de guiarme para llevar a buen puerto el proyecto.

Tutor del proyecto y Comercial: Jeremy Romano es mi tutor del proyecto, es el Ingeniero Comercial por tanto quien habla con el cliente y nos dijo las necesidades de éste.

[3]http://www.cs.waikato.ac.nz/ml/weka/
[4]http://scikit-learn.org/stable/

Cliente: El cliente es la empresa Orange que ha encargado que se haga el prototipo a mi empresa.

2 Estado del arte

2.1 Elasticsearch

Para llevar a cabo este proyecto es necesario utilizar el servidor de búsqueda Elasticsearch[7] dado es dónde la empresa Orange tiene todos sus datos. Así pues en esta sección hablaremos brevemente de esta tecnología.

Elasticsearch es un servidor de búsqueda basado en Apache Lucene. Permite la búsqueda de toda clase de documentos y los almacena en formato JSON en un índice. Esto permite que se pueda almacenar todo tipo de ficheros mientras se pueda extraer texto de él. Es un sistema distribuido que funciona con nodos que guardan fragmentos del índice así como replicas en caso de que uno de los nodos se cuelgue. Esta desarrollado en Java y es de código abierto.

En el mercado encontramos otras tecnologías similares como Apache Solr sin embargo Elasticsearch parece situarse a la cabeza como opción más utilizada[8]. Además como hemos dicho la empresa tiene sus datos con Elasticsearch.

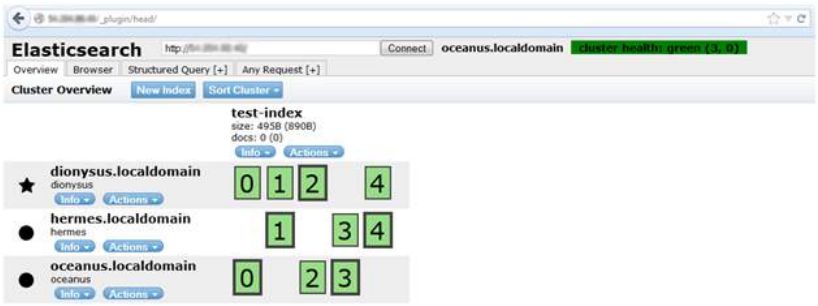

Figure 1: Ejemplo de un índice y sus nodos en Elasticsearch.

Para facilitarnos la entrada de datos en Elasticsearch, se nos proporciona otra herramienta también desarrollada por los que hacen Elasticsearch llamada Logstash. Esta herramienta realiza un tratamiento de los logs basados en reglas que nosotros imponemos y manda los datos directamente a Elasticsearch para que sean indexados y buscados. Podemos definir el formato

que queremos del JSON resultante así como que partes coger del documento o log de entrada. Hay una gran variedad de ficheros ya hechos para logs tipo como los de Apache que nos facilitan el trabajo.

Finalmente otra herramienta de esta suite que voy a utilizar es Kibana. Kibana proporciona una interfaz gráfica para visualizar y crear gráficos de todo tipo de los datos almacenados en Elasticsearch. Nos permite entender mejor nuestros datos al tener la facilidad de crear visualizaciones y analizar estos.

Figure 2: Aquí vemos como Kibana nos facilita el trabajo gracias a su interfaz gráfica sencilla y que permite crear gráficos de nuestros datos.

2.2 Clustering

El Clustering (o algoritmo de agrupamiento)[9] consiste en agrupar una serie de vectores según un criterio en grupos o clusters. Generalmente el criterio suele ser la similitud por lo que diremos que agrupa los vectores similares en grupos[10]. Está considerado como un aprendizaje no supervisado dentro de la minería de datos.

En esta sección vamos a explicar brevemente algunos algoritmos de clustering.

2.2.1 K-means

Este algoritmo particiona los N objetos en K particiones (K siendo un valor arbitrario) en donde un objeto irá al cluster con la media más cercana. El algoritmo asigna K centros aleatoriamente, luego asigna los objetos al centro más cercano. El centro se recalcula como la media de los puntos que tiene asignado, una vez actualizado se vuelven a reasignar los objetos al más cercano y así hasta tener convergencia [11].

Este algoritmo es NP-Hard. Depende mucho de la asignación inicial de los centros, nos puede dar un resultado u otro por lo que es mejor hacer varias pruebas con diferentes valores. Una variante llamada K-means++ [12] intenta resolver este problema al escoger mejores centros.

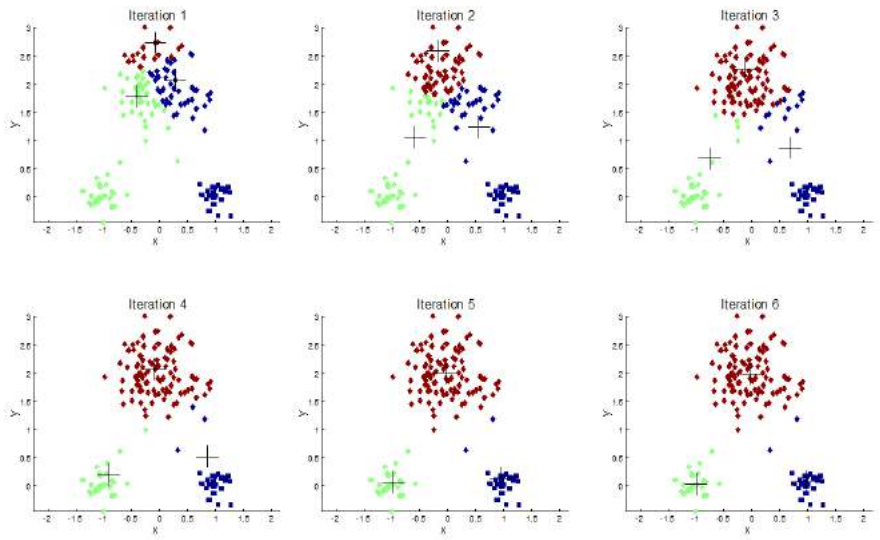

Figure 3: Todas las iteraciones del algoritmo K-means.

En la primera iteración vemos tres cruces correspondientes a los 3 centros iniciales. Se han asignado los puntos al centro más cercano, es visible gracias a los colores. Tras la asignación, se calcula la media de los puntos asignados al cluster y se actualiza el centroide con este valor. Vemos en la

iteración dos como se han desplazados los centroides y se han reasignado los puntos al centro más cercano. Se sigue realizando estos pasos hasta que las asignaciones de los puntos no cambien.

Existen una variedad de heurísticas para que el algoritmo converja rápidamente. La principal desventaja de este algoritmo es que hay que especificar el numero de clusters que queremos que nos forme, y dependiendo de este numero los resultados varían.

2.2.2 K-nearest neighbors

En este algoritmo se decide la membresia de un objeto teniendo en cuenta sus vecinos. Se decide a que cluster pertenece mirando a que cluster pertenece la mayoría de sus vecinos K más cercanos a el. Se trata de unos de los algoritmos más simples de aprendizaje automático[13].

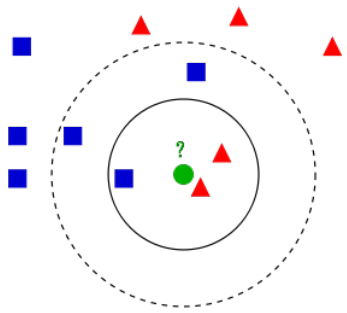

Figure 4: El circulo verde debería ser clasificado con los azules o rojos según el numero K de vecinos que escojamos. Si k=3 será asignado al rojo ya que de sus 3 vecinos más cercanos dos son rojos. Si k=5 irá al azul al tener tres vecinos azules y dos rojos.

2.2.3 DBSCAN

El DBSCAN [14] es un algoritmo que se basa en la densidad para realizar la clasificación. Hay que fijar un radio E en el cual queremos encontrar puntos y un numero mínimo de puntos P que se ha de encontrar dentro del radio.

1. Partimos de un punto aleatorio y miramos si desde ese punto hay un mínimo de puntos P en un radio de distancia E.

2. Si es así, esto forma un cluster y volvemos a la etapa 1 con un punto de los encontrados.

3. Si no encontramos el mínimo de puntos pero hemos llegado a ese punto a través de un punto que si lo cumplía, éste formará parte del cluster. En caso de no poder llegar a un punto a través de otros no formará parte del cluster y será un nodo ruidoso.

Este algoritmo no hay que fijar el numero de clusters y tiene una complejidad de $O(n \ \log n)$.

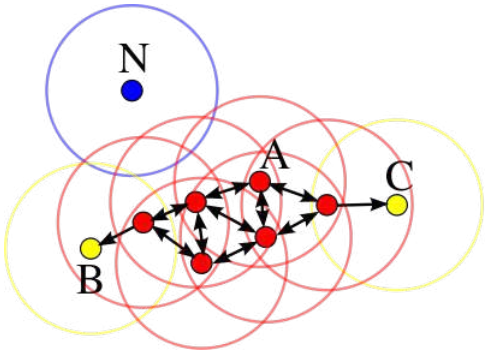

Figure 5: En este caso el mínimo numero de puntos es P = 3. Los puntos de A todos contienen al menos 3 puntos en un radio E y por ello forman un cluster. Tanto B como C son alcanzables desde A por lo que formarán parte del cluster, son el limite de éste. N es un nodo ruidoso, no es alcanzable por ningún punto.

2.3 Aprendizaje supervisado

El aprendizaje supervisado es una técnica para deducir una función a partir de datos de entrenamiento. Se entrena un modelo con ejemplos predefinidos y luego se usa ese modelo para predecir los resultados de los datos de entrada que le pasemos.

A continuación vamos a ver brevemente algunos ejemplos de algoritmos de aprendizaje supervisado.

2.3.1 Support vector machine

Dado un conjunto de ejemplos de entrenamiento, una SVM los representa como puntos en el espacio, separando las clases por la mayor distancia posible. Cuando las nuevas muestras se ponen en correspondencia con dicho modelo, según su proximidad serán asignadas a una u otra clase. Es decir, si un punto nuevo pertenece a una categoría o la otra[15].

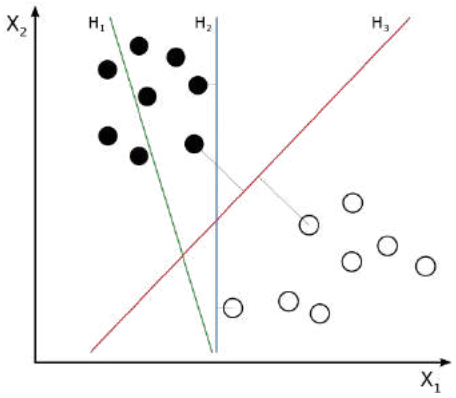

Figure 6: H1 no separa las clases. H2 si pero H3 lo hace con la separación máxima.

2.3.2 Naive Bayes classifier

Es un clasificador probabilístico basado en el teorema de Bayes y algunos hipótesis simplificadoras. Asume que la presencia o ausencia de una característica particular no está relacionada con la presencia o ausencia de cualquier otra característica, dada la clase variable. Se requiere una pequeña cantidad de datos de entrenamiento para estimar los parámetros necesarios para la clasificación[16].

2.3.3 Decision Tree Learning

Utiliza un árbol de decisión como modelo predictivo que mapea observaciones sobre un artículo a conclusiones sobre el valor objetivo del artículo. Las hojas representan etiquetas de clase y las ramas las características que conducen a

esa clase. Cada nodo representa uno de los valores de entrada y sus nodos hijos representan un posible valor para esa variable de entrada. Las hojas representan un posible valor para la entrada dada[17].

3 Análisis de texto

Llegados a este punto podemos preguntarnos como aplicar estos algoritmos, que realizan operaciones matemáticas, a documentos que contienen palabras. ¿ Cómo es posible saber si un documento es similar a otro? Estas preguntas vamos intentar responderlas en los siguientes apartados.

3.1 Similitud coseno

Se utiliza en la minería de datos para saber la semejanza que hay entre documentos. En concreto, un algoritmo de clustering como los vistos anteriormente, utilizaría la similitud coseno para ver la semejanza entre documentos y formar los clusters.

La similitud coseno es una medida de la similitud existente entre dos vectores en un espacio que posee un producto escalar con el que se evalúa el valor del coseno del ángulo comprendido entre ellos. Esta función trigonométrica proporciona un valor igual a 1 si el ángulo comprendido es cero, es decir si ambos vectores apuntan a un mismo lugar. Cualquier ángulo existente entre los vectores, el coseno arrojaría un valor inferior a uno. Si los vectores fuesen ortogonales el coseno se anularía, y si apuntasen en sentido contrario su valor sería -1. De esta forma, el valor de esta métrica se encuentra entre -1 y 1, es decir en el intervalo cerrado [-1,1]. Muchas veces se presenta el valor de manera que esté entre [0,1].

3.2 Transformar un documento a valores numéricos

La transformación de un documento, que contiene palabras, a un vector de números con el cual los algoritmos pueden trabajar se realiza de la siguiente manera. Supongamos estos tres documentos (cada frase es un documento):

1. Juega al fútbol.
2. Le gusta el baloncesto.
3. Mira un partido de fútbol.

La siguiente matriz tiene como filas a los documentos, y una columna por cada palabra diferente que hay en el total de documentos (vocabulario). La

idea es poner la frecuencia de la palabra en el documento para cada palabra. De esta manera el documento nº1 tiene las palabras "juega","al" y "fútbol" por lo que la fila uno tiene valor 1 para esas palabras y 0 para el resto.

	juega	al	futbol	le	gusta	el	baloncesto	mira	un	partido	de
1	1	1	1	0	0	0	0	0	0	0	0
2	0	0	0	1	1	1	1	0	0	0	0
3	0	0	1	0	0	0	0	1	1	1	1

Solo con mirar las tres frases, uno diría que la 1 y la 3 son similares dado que ambas hablan de fútbol. Sin embargo, si aplicamos la similitud coseno entre el 1 y el 3 nos da un valor de 0.258, recordemos que 0 es no se parecen nada y 1 los documentos son idénticos. Entre el 1 y el 2 nos da similitud = 0 y entre el 2 y el 3 también 0. Por lo que en este ejemplo tan simple parece que funcionan bien estas técnicas.

Hay aun ciertas cosas que pulir en este método, para empezar hay palabras que van a salir con frecuencia en muchos documentos. Los artículos (la,el,ella,...) o preposiciones (a,de,por,...) van a estar en gran cantidad en los documentos que analicemos. Realmente no nos dan ninguna información sobre el documento o similitud con otros, volviendo al ejemplo anterior, la palabra que era clave para la similitud era "fútbol". Hay que eliminar palabras superfluas que solo nos hacen ocupar espacio en la matriz. Este tipo de palabras se les denomina "stopwords". Antes de empezar a trabajar con los documentos conviene eliminar las stowords de los documentos, nosotros usaremos una lista que ya incluye la librería NLTK para Python[5]. Ignoraremos toda palabra que aparezca en el documento que esté en esa lista de stopwords.

Pero,¿qué pasaría si todos nuestros documentos tratasen sobre fútbol? Ciertamente, todos contendrían la palabra "fútbol" y varias veces, por lo que esta palabra realmente tampoco nos ayuda mucho a calcular la similitud entre los documentos. Es una palabra que por aparecer en todos los documentos, ya no es significativa. Vamos a ver como resolver esto.

[5]http://www.nltk.org/

3.3 Tf-IDF

Tf-Idf son las siglas de Term frequency – Inverse document frequency (frecuencia de término - frecuencia inversa de documento) es una medida numérica que expresa cuan relevante es una palabra para un documento en una colección. El valor de Tf-idf aumenta proporcionalmente a la frecuencia de una palabra (de ahí TF) pero es compensada por la frecuencia de esa palabra en la colección de documentos, lo que permite manejar el hecho de que algunos palabras son más comunes en la colección.

Supongamos que tenemos una colección de documentos que hablan sobre todos los equipos de la liga de fútbol. Está claro que el termino fútbol aparecerá con frecuencia y se encontrará en todos los documentos. Por tanto no es un termino relevante para distinguir documentos. El hecho de que aparezca mucho, ha de ser compensado por el hecho de que aparece en todos los documentos. En cambio términos que solo aparecen en uno o pocos documentos tendrán más valor ya que definen esos documentos con respecto a los otros. Por eso se incorpora un factor de frecuencia inversa de documento que atenúa el peso de los términos que ocurren con mucha frecuencia en la colección de documentos e incrementa el peso de los términos que ocurren pocas veces.

Como Tf-idf es el producto de dos medidas, vamos a mostrar por separado su formulación.

La frecuencia de termino (TF) viene dado por la frecuencia de un término t en un documento d dividido por la frecuencia del término más frecuente (máximo) en el documento:

$$\mathrm{tf}(t,d) = \frac{\mathrm{f}(t,d)}{\max\{\mathrm{f}(w,d) : w \in d\}}$$

La frecuencia inversa de documento:

$$\mathrm{idf}(t,D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}$$

donde:

$|D|$: cardinalidad de D, o número de documentos en la colección.

$|\{d \in D : t \in d\}|$: número de documentos donde aparece el término t.

3.4 Stemming

El stemming es un método para reducir una palabra a su raíz. Stemming aumenta el recall que es una medida sobre el número de documentos que se pueden encontrar con una consulta.

Cuando queremos buscar una palabra en documentos, tal vez nos interese utilizar este método para recuperar más documentos relacionados con la palabra. Por ejemplo si buscamos "biblioteca" a secas, nos devolverá los documentos que contengan esta palabra. pero si aplicamos stemming, la raíz seria "bibliotec" y nos devolvería documentos que contengan además de "biblioteca" los que tienen "bibliotecario".

3.5 Lematización

La lematización es un proceso lingüístico que consiste en, dada una forma flexionada (es decir, en plural, en femenino, conjugada, etc), hallar el lema correspondiente. El lema es la forma que por convenio se acepta como representante de todas las formas flexionadas de una palabra.

Por ejemplo : "dije","decían","diré" tienen como lema la palabra "decir".

4 Ámbito del proyecto

En esta sección hablaremos del ámbito del proyecto y sus objetivos. Describiremos la problemática y los pasos que seguiremos para resolverla.

4.1 Objetivos

Hay varios objetivos en la realización de este proyecto. Primeramente hay una parte de familiarizarse con las tecnologías a utilizar, en este caso ElasticSearch[6], Kibana[7] y Logstash[8] .

Seguidamente, hay que realizar un estudio/investigar sobre los algoritmos de clustering y las diferentes herramientas que nos pueden ayudar a resolver la problemática, todo ello ha de ser utilizando tecnologías Open-source.

Luego hay que empezar una fase de recolección de datos de Twitter. Deberemos recoger tweets que hable sobre Orange y realizar un filtrado y limpieza de los datos.

Hay que separar los tweets que hablan de la empresa Orange y los que se refieren a otros temas. Utilizaremos algoritmos de aprendizaje automático por lo que hay que primero manualmente crear unos datos de entrenamiento, es decir tener unos datos ya etiquetados sobre los que entrenar el modelo. También deberemos crear uno para el análisis de sentimiento.

Finalmente, aplicaremos algoritmos de clustering a los datos para ver que información podemos extraer.

4.2 Alcance

Debemos hacer un estado del arte de los diferentes métodos de clustering que hay. Estudiar que tipo de información podemos extraer, como implementar los algoritmos y las ventajas e inconvenientes de éstos.

[6]https://www.elastic.co/
[7]https://www.elastic.co/products/kibana
[8]https://www.elastic.co/products/logstash

Hay que probar diferentes herramientas y librerías de clustering. Actualmente encontramos un gran numero de herramientas open source que nos permiten hacer clustering de documentos tales como Weka, Carrot2[9] y Orange[10].

Además, debemos decidir en que lenguaje de programación vamos a implementar el prototipo. Deberemos para ello realizar comparativas de los lenguajes mas indicados para el data mining y de sus librerías. En Pyhton tenemos la librería Scikit-learn y R nos proporciona en su librería de data mining métodos y algoritmos que nos ayudarán con el clustering.

4.3 Obstáculos del proyecto

En esta sección analizaremos los posibles obstáculos que nos podemos encontrar en la realización del proyecto así como sus posibles soluciones.

4.3.1 Evaluación de métodos de Clustering

Para saber que tan bueno es un algoritmo de aprendizaje supervisado es bastante fácil, dado que los datos tienen etiquetas predefinidas y el algoritmo solo ha de etiquetar nuevos datos basándose en datos ya etiquetados.

Pero los algoritmos de clustering lo que hacen es formar clusters (grupos) de manera que los objetos en cada grupo sean similares entre ellos. ¿Cómo podemos saber si un objeto debía pertenecer a un grupo u otro ? Bien podría pertenecer a más de un cluster y hay algoritmos que permiten esto. Es por tanto un criterio más subjetivo y evaluar que tan bueno son los clusters no es fácil.

Si bien existen métodos para evaluar un algoritmo de clustering[18], y que podrían ser utilizados en el proyecto, lo más típico es que un humano evalúe los clusters. Por ejemplo si se hacen clustering sobre artículos, si se cogen artículos de fútbol, baloncesto y tenis, podríamos hacer tres clusters y ver si forma bien los clusters separados por deporte.

[9] http://project.carrot2.org/
[10] http://orange.biolab.si/

Hay que tener en cuenta que es posible no obtener los resultados esperados. Se trata de hacer también un estudio para ver si es posible aplicar clustering pero quizás nos encontremos con que los métodos no sirven para este problema.

4.3.2 Elección de herramientas Open-source

Dentro de los objetivos estaba el de utilizar tecnologías Open-source. Esto es puesto que siempre que es posible , desde de un punto de vista de la empresa, utilizar tecnologías que no supongan coste alguno.

Debemos pues realizar una búsqueda de las distintas herramientas y algoritmos que hay y escoger aquellos que convengan. Para saber cual nos conviene debemos probar estas herramientas o algoritmos con las mismas condiciones y medir el uso de memoria y el tiempo de ejecución del algoritmo pues queremos el más eficiente.

5 Metodología

A continuación hablaremos de las herramientas de desarrollo que vamos a utilizar y de la metodología que vamos a seguir.

5.1 Herramientas de desarrollo

Ya hemos comentado que vamos a utilizar Elasticsearch junto con Logstash y Kibana. Estas herramientas ya han sido presentadas por lo que no hace falta decir más.

Como herramientas de desarrollo vamos a utilizar varias dado que vamos a tratar diferentes lenguajes de programación. Para el lenguaje de programación R utilizaremos el *RStudio* con diferentes librerías instaladas. Para Python usaremos *Sublime Text 2* con plugins.

Para utilizar Elasticsearch vamos a instalar varios plugins que nos facilitarán el trabajo como el plugin *Head* o *Sense*. Además utilizaremos una maquina virtual linux con Virtualbox.

Para los programas hechos en Python utilizaremos el conjunto de herramientas Scikit-learn y para la interfaz gráfica *Tkinter*.

Usaremos la herramienta *Weka* que contiene una colección de algoritmos de aprendizaje automático y tratamiento de textos. Tanto la versión gráfica como el código en Java. Para Java usaremos el IDE *Eclipse*.

Utilizaremos el *API de Twitter* para poder realizar búsquedas en Twitter y recuperar los resultados para luego extraer información que nos pueda interesar.

Finalmente, para hacer un seguimiento de los cambios del proyecto usaremos la herramienta *Git*.

5.2 Seguimiento del proyecto

Junto con el jefe de proyecto hacemos reuniones periódicamente para ver el estado del proyecto, como avanza y hacia donde vamos.

Así mismo *Git* nos permitirá hacer el seguimiento del código del programa y sus cambios.

5.3 Método de validación

Debido a que vamos a utilizar diferentes algoritmos y métodos, algunos en los cuales no se puede saber que tan bueno es un resultado, vamos a explicar en cada caso como se va a validar el resultado.

Para los algoritmos de aprendizaje supervisado, en el que el modelo nos da un resultado, tenemos lo que se conoce como test de validación. Un test de validación nos permitirá afinar los parámetros del algoritmo para obtener el resultado esperado. En este caso el test de validación no es mas que una serie de muestras ya clasificadas previamente en las que probar el modelo.

Una forma de estimar la precisión de un modelo es coger los datos de muestra y hacer dos subconjuntos: uno seria los datos de entrenamiento y otro los datos de prueba. Los datos de entrenamiento se utilizará para entrenar el modelo y los de prueba para validarlo. Entrenamos el algoritmo y luego probamos el modelo sobre los datos de prueba, que al estar ya etiquetados podemos saber la precisión del modelo. Sin embargo, este método depende demasiado de como hayan sido hechas las particiones por lo que no es tan fiable.

5.3.1 Validación cruzada

La validación cruzada (en inglés Cross-Validation) es una técnica para evaluar los resultados y garantizar que son independientes de la partición entre datos de entrenamiento y de prueba.

La validación cruzada realiza varias iteraciones cambiando cada vez el subconjunto de entrenamiento y de pruebas. Se hace la media aritmética de los resultados de cada iteración y ese será el resultado final. De esta manera nos aseguramos que no tendremos un resultado que es dependiente de los

subconjuntos de entrenamiento y pruebas que hemos escogido.

Nosotros utilizaremos la validación cruzada de K iteraciones. Los datos de muestra se dividen en K subconjuntos. Uno de los subconjuntos se utiliza como datos de prueba y el resto (K-1) como datos de entrenamiento. El proceso de validación cruzada es repetido durante K iteraciones, con cada uno de los posibles subconjuntos de datos de prueba. Se hace la media aritmética de los resultados para obtener el resultado final.

6 Planificación Temporal

En esta sección vamos a tratar de explicar la planificación temporal del proyecto. Explicaremos las diferentes tareas a realizar y cuanto tiempo calculamos que nos tomará. Así mismo mostraremos un diagrama de Gantt de las tareas del proyecto.

En mi caso estoy en una modalidad en la que hago practicas en una empresa. Las practicas comenzaron el 15 de Junio de 2015 y finalizan el 15 de Diciembre de 2015 lo que hace un total de 6 meses.

6.1 Planificación del proyecto

Esta es la tarea que hacemos actualmente y consta de las siguientes partes:

1. Ámbito del proyecto
2. Planificación temporal
3. Gestión económica y sostenibilidad
4. Estado del arte

6.2 Descripción de las tareas

La realización de este proyecto consta de dos grandes etapas. La primera consiste en realizar un estudio y familiarizarse con las tecnologías y algoritmos, digamos un estado del arte del clustering. Hay que documentar las ventajas e inconvenientes de los métodos así como de las herramientas utilizadas y librerías.

Una vez hecho el estudio y haberse familiarizado con las diferentes herramientas, queda reunirse con el cliente para hablar en detalle sobre el problema a resolver. El cliente es Orange y quieren realizar un estudio sobre frameworks de aprendizaje automático y algoritmos de clustering.

La siguiente lista resume las tareas del proyecto:

1. Familiarizarse con la tecnología Elasticsearch + Kibana + Logstash

2. Estado del arte del clustering y aprendizaje supervisado.
3. Benchmark de herramientas y lenguajes de programación
4. Reunión con el cliente y análisis del problema
5. Recolección datos y limpieza
6. Clustering y análisis de sentimiento
7. Redacción de la memoria y documentación.

6.3 Hito final

En el hito final se realizará la presentación del TFG delante del tribunal. Además de entregar la memoria junto con el código a la Universidad también he de redactar un informe para la empresa que trabajo con todo lo que he hecho y las conclusiones que he sacado. Finalmente hay que presentar el prototipo y las conclusiones a la empresa Orange para que ellos evalúen si es viable y les sale a cuenta implementarlo o por el contrario no les sirve.

6.4 Tabla de tareas

Tarea	**Duración (en horas)**
Familiarizarse con Elasticsearch	230
Estado del arte del clustering	100
Benchmark de herramientas	80
Reunión con el cliente y análisis del problema	20
Recolección de datos	50
Clustering y análisis de sentimiento	270
Hito final	90
Total	**840**

6.5 Diagrama de Gantt

Task Name	Start Date	End Date	Duration
Familiarizarse con tecnologias	**06/15/15**	**07/20/15**	**26d**
ElasticSearch + Kibana + Logstash	06/15/15	07/20/15	26d
Estado del arte del clustering	**07/21/15**	**08/13/15**	**18d**
Investigar sobre distintos metodos	07/21/15	07/27/15	5d
Estudiar dichos metodos	07/28/15	08/07/15	9d
Realizar informe	08/10/15	08/13/15	4d
Bench de herramientas	**08/11/15**	**08/31/15**	**15d**
Weka	08/11/15	08/17/15	5d
Scikit-learn	08/17/15	08/26/15	8d
RStudio	08/26/15	08/31/15	4d
Carrot2			
Analisis sentimental Twitter	09/01/15	09/30/15	22d
Analisis del problema	09/01/15	09/08/15	6d
Estudio de algoritmos	09/08/15	09/18/15	9d
Test algoritmos	09/18/15	09/30/15	9d
Reunion con Orange	11/02/15	11/02/15	1d
Implementación del Prototipo	11/03/15	12/11/15	29d
Hito final	12/11/15	12/23/15	9d

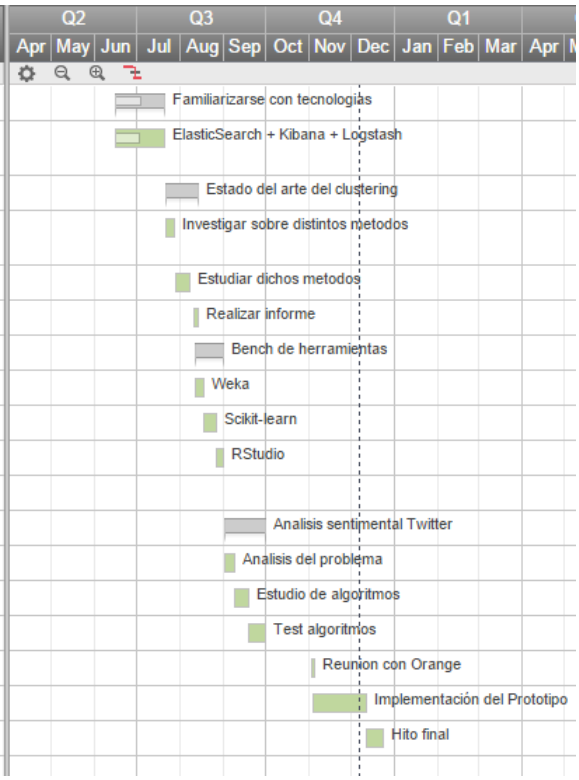

Figure 7: Diagrama de Gantt de las tareas.

6.6 Recursos

Los recursos que utilizaremos para llevar a cabo este proyecto tanto hardware como software se presentan a continuación.

6.6.1 Hardware

Como maquina utilizo un Lenovo T530 con las siguientes caracteristicas:

- Procesador Intel Core i5-3320M CPU @ 2.60 GHz
- Memoria RAM 4 GB

- Disco Duro SATA de 320 GB
- Tarjeta Gráfica NVIDIA® Optimus™
- Pantalla de 15.6" HD
- Un segundo monitor Lenovo ThinkVision de 19"
- Teclado y ratón Logitech

Además hemos hecho uso de otros dispositivos como webcams, proyectores o pantallas de TV.

6.6.2 Software

Las herramientas de software utilizadas son:

- Windows 7 Profesional 64 bits
- Debian 8 (Jessie)
- Oracle VM Virtualbox
- Scikit-learn
- Sublime Text 2
- Eclipse
- RStudio
- Python console
- Weka
- API de Twitter

7 Valoración de alternativas y plan de acción

Hay una gran variedad de algoritmos de clustering por lo que si un algoritmo no cumple nuestras necesidades podemos probar otro. Además, como hemos dicho, también existe la posibilidad de utilizar el aprendizaje automático, con lo que en este sentido tenemos una gran variedad de algoritmos y creo que podemos encontrar alguno que nos convenga.

Afortunadamente también hay muchos frameworks open-source orientados al aprendizaje automático y clustering por lo que podemos explorar varias posibilidades sino nos convencen. Ya hemos nombrado algunos de los que íbamos a probar anteriormente.

Como es una temática algo novedosa para mi, en la empresa Orange hay alguien especializado en machine learning que me podrá guiar y ayudar con posibles dudas que tenga sobre los algoritmos. Además, en mi empresa cuento con la ayuda del Jefe de proyecto que tiene experiencia o los demás compañeros si hiciera falta.

Al haber una parte de investigación, y un tiempo previsto para ello que es considerablemente holgado, no creo que haya problemas en los tiempos luego para implementar el prototipo. De hecho por lo que he estado viendo las primeras semanas, voy bastante adelantado a los plazos. También el hecho de haber una parte previa de Clustering de los logs antes de implementar el prototipo nos da margen para estudiar que algoritmo aplicar y hacer diferentes pruebas antes de empezar el desarrollo del prototipo.

Las diferentes partes del proyecto están bien definidas y se complementan entre ellas de manera que la siguiente está relacionada con la anterior. Se aprovecha pues el trabajo de cada etapa en la siguiente y dividiendo así las etapas nos aseguramos que el proyecto vaya avanzando.

8 Gestión económica

En este apartado trataremos la parte económica del proyecto, veremos los costes de recursos humanos, software y hardware.

8.1 Recursos Humanos

El proyecto será llevado a cabo por una persona a excepción del jefe de proyecto:

Rol	Horas	Precio por hora	Precio total
Jefe de proyecto	120	55 €	6600 €
Analista	200	37 €	7400 €
Desarrollador	330	30 €	9900 €
Tester	190	26 €	4940 €
Total	**840**		**28840 €**

8.2 Tareas

Tarea	Horas	Coste
Familiarizarse con Elasticsearch	230	6700 €
Estado del arte del clustering	100	3400 €
Benchmark de herramientas	80	3000 €
Análisis del problema	20	1500 €
Recolección y filtrado de datos	100	5800 €
Implementación algoritmos	220	7000 €
Hito final	90	2300 €
Total	**840**	**29700 €**

8.3 Presupuesto hardware

Para llevar a cabo el proyecto necesitamos una estación de trabajo que incluya un ordenador con acceso a internet. La siguiente tabla muestra los costes de la estación de trabajo:

Producto	Precio	Unidades	Vida util	Amortización
Lenovo ThinkPad T530	1500 €	1	4 años	375 €
Lenovo ThinkVision Monitor	290 €	1	5 años	58 €
Ratón y teclado Logitech	45 €	1	5 años	9 €
Total	**1835 €**			**442 €**

8.4 Presupuesto software

La siguiente tabla vemos los costes del software que vamos a utilizar para llevar a cabo el proyecto:

Producto	Precio	Unidades	Vida util	Amortización
Windows 7 Professional	130 €	1	4 años	32,5 €
MS Office 2007	250 €	1	7 años	7 €
Eclipse	0 €	1	3 años	0 €
Debian 8.0	0 €	1	3 años	0 €
Scikit-learn	0 €	1	3 años	0 €
Weka	0 €	1	3 años	0 €
Sublime Text 2	0 €	1	3 años	0 €
LaTeX	0 €	1	3 años	0 €
Total	**280 €**			**39,5 €**

8.5 Otros costes

Producto	Precio	Total tiempo	Precio total
Electricidad	0,14670 €/kWh	840 h	123,228€
Internet	35 €/mes	6 meses	210 €
Total			**333,228 €**

8.6 Resumen de costes

Concepto	Costes
Recursos Humanos	29700 €
Hardware	442 €
Software	39,5 €
Otros costes	333,228 €
Total	**30514,728 €**

9 Sostenibilidad

Aquí hablaremos del impacto que tiene el proyecto a nivel económico, social y sostenible. Al final, haremos una valoración de la viabilidad y sostenibilidad del proyecto.

9.1 Dimensión económica

Se ha realizado un estudio de los costes del proyecto para comprobar si el proyecto es viable económicamente. Hay que decir que este proyecto entra dentro del departamento de I+D. Típicamente en este tipo de departamentos se hacen pruebas y experimentos sin pensar demasiado en la viabilidad económica inmediata.

Es por ello que el proyecto pueda no dar beneficios o al menos beneficios inmediatos pero siempre hay que explorar e innovar en este mundo y por ello es necesario ver si se pueden aplicar nuevas tecnologías. Si el proyecto da buenos resultados el beneficio será grande ya que se ahorrará tiempo de calculo y por ello recursos utilizados por los servidores de Orange.

En este proyecto trabajan dos personas, incluido el Jefe de proyecto que se encarga de guiar el proyecto y controlarlo. Los demás roles son llevados a cabo por una única persona por lo que no supone mucho coste para la empresa. Los posibles beneficios no son solo a nivel económico, sino demostrar que se esta a la vanguardia de las nuevas tecnologías y que se innova dentro de la empresa.

En conclusión, es un proyecto que es totalmente asumible por la empresa y los beneficios pueden ser económicos pero también a largo plazo la imagen de la empresa se ve fortalecida.

9.2 Dimensión social

Se trata de un proyecto para el buscador de Orange. Ya hemos dicho que es el 4º a nivel de cuota de usuarios en Francia, representa un 0,4% de cuota de mercado.

Los buscadores facilitan mucho la vida ya que gracias a ellos podemos buscar cualquier información que haya en Internet. Han cambiado drásticamente como utilizamos Internet, aunque yo siempre he conocido Internet con buscadores como Google, está claro que hubo un antes y un después de la aparición de los buscadores.

El motor de búsqueda de Orange no tiene el impacto de Google, aunque hay que decir que es anterior a este y en los últimos tiempos, Orange ha incluido funciones que luego Google ha copiado para su buscador.

Espero poder contribuir a este motor de búsqueda que tiene casi 20 años de historia y que todavía sigue siendo utilizado por parte de un gran numero de usuarios.

9.3 Dimensión ambiental

Uno de los objetivos del proyecto era utilizar tecnologías ya existentes dado que ahorra trabajo y costes. Utilizaremos herramientas y algoritmos Open-source por lo que aprovechamos el trabajo hecho por otros.

A nivel de recursos se utilizará un ordenador portátil y un monitor, que obviamente utilizan electricidad pero el impacto ambiental es el esperado, no se puede realizar el trabajo sin estos recursos.

Finalmente, el trabajo la idea es que optimice el buscador, por lo que ahorrará recursos energéticos a la empresa al ahorrar cálculos innecesarios y también facilitará las búsquedas a los usuarios teniendo un doble impacto.

9.4 Conclusiones

Sostenible	Económica	Social	Ambiental
Planificación	Viabilidad económica	Mejora en calidad de vida	Análisis de recursos
Valoración	10/10	7.5/10	8/10

10 Adecuación a la especialidad

Mi especialidad es Computación y en esta sección justificaremos porque el proyecto es valido para ésta especialidad.

10.1 Asignaturas relacionadas con la especialidad

Las siguientes asignaturas tienen relación con lo que voy a hacer en el proyecto:

- **Algorísmia (A) :** Esta asignatura es fundamental ya que en ella vimos una gran cantidad de algoritmos y estructuras de datos. Nos ayudará saber como calcular el coste computacional y como mejorar un algoritmo para que sea eficiente.
- **Intel·ligència Artificial (IA) :** En esta asignatura trabajamos con algoritmos de busqueda en grafos y también vimos un poco de aprendizaje automatico, que será muy importante en este proyecto.
- **Llenguatges de Programació (LP) :** Aprendimos el lenguaje de programación Python que será muy utilizado en la realización del proyecto.
- **Cerca i Anàlisi d'Informació Massiva (CAIM) :** Quizás la que mas relación tiene con el proyecto, utilizamos Lucene que es la base de ElasticSearch, vimos la matriz Tf-idf que nos será util para tratar documentos y también vimos algoritmos de clustering.

10.2 Justificación

Se trata de utilizar algoritmos de clustering y aprendizaje automático. Dado la naturaleza del proyecto de buscar optimizar el motor de búsqueda está claro que entra dentro de la especialidad de Computación. Uno de los criterios más importantes del proyecto es la eficiencia de los algoritmos y su fiabilidad, que es también un aspecto importante de la especialidad de Computación.

El prototipo tiene que hacer un uso eficiente de los recursos por lo que habrá que elegir bien las estructuras de datos que vamos a utilizar. También debemos optimizar al máximo posible su ejecución.

Entra dentro de las competencias de la especialidad el saber escoger que algoritmo es el adecuado, el hacer un uso eficiente de los recursos y en general los algoritmos de aprendizaje. Por todos estos motivos este proyecto entra dentro de las competencias de Computación.

10.3 Competencias técnicas

- **CCO1.1:** Avaluar la complexitat computacional d'un problema, conèixer estratègies algorísmiques que puguin dur a la seva resolució, i recomanar, desenvolupar i implementar la que garanteixi el millor rendiment d'acord amb els requisits establerts. [Bastant]
- **CCO1.3:** Definir, avaluar i seleccionar plataformes de desenvolupament i producció hardware i software per al desenvolupament d'aplicacions i serveis informàtics de diversa complexitat. [Una mica]
- **CCO2.3:** Desenvolupar i avaluar sistemes interactius i de presentació d'informació complexa, i la seva aplicació a la resolució de problemes de disseny d'interacció persona computador. [Una mica]
- **CCO2.4:** Demostrar coneixement i desenvolupar tècniques d'aprenentatge computacional; dissenyar i implementar aplicacions i sistemes que les utilitzin, incloent les que es dediquen a l'extracció automàtica d'informació i coneixement a partir de grans volums de dades. [Bastant]
- **CCO2.5:** Implementar software de cerca d'informació (information retrieval). [En profunditat]
- **CCO3.1:** Implementar codi crític seguint criteris de temps d'execució, eficiència i seguretat. [Bastant]

11 Benchmark de Herramientas

Hemos realizado un estudio sobre los dos principales frameworks de machine learning para comparar la eficiencia y el uso de memoria. Por una parte tenemos Weka[4] que está programado en Java y por otra Scikit-learn en Python pero los algoritmos clave están programados en C para mayor eficiencia.

Para comparar, hemos cogido dos algoritmos y hemos utilizado el mismo dataset para la versión del algoritmo de Weka y de Scikit-learn , aunque para cada algoritmo el dataset es distinto. Cada algoritmo se ha ejecutado 10 veces, los valores son la media de esas 10 ejecuciones, a continuación una breve comparativa:

11.1 K-means

Framework	Tiempo ejecución (s)	Memoria (MB)
Scikit-learn	0.19203	1.5882
Weka	0.5378	7.6768

11.2 Multinomial Naive Bayes

Al ser un algoritmo de aprendizaje automatico, hemos medido el tiempo y memoria utilizada para el aprendizaje y para el test.

Framework	T. apren. (s)	M. apre. (MB)	T. test (s)	M. test (MB)
Scikit-learn	0.0026	0.0632	0.00044	0.02344
Weka	0.0738	8.7195	0.0318	7.3585

De estos tests sacamos la conclusión de que Scikit-learn es más eficiente y por tanto es el framework elegido.

12 Recolección de datos de twitter

12.1 Introducción

Twitter es una popular red social fundada en San Francisco en 2006. Su principal caracteristica es que los mensajes tienen un limite de 140 caracteres, son los llamados "tweets". Se estima que tiene mas de 800 millones de usuarios en todo el mundo y que genera 65 millones de tweets al dia[11].

Los usuarios pueden seguir a otros usuarios (suscribrse a sus tweets) y cada vez que alguien escribe un tweet, se puede retwitear, es decir, compartir el mensaje con tus seguidores para darle mas visibilidad. También existen los denominados "hashtags", que son palabras o frases antecedidas por un "#" para agrupar mensajes comunes sobre un mismo tema.

Los motivos por los cuales vamos a elegir Twitter sobre otra red social son sencillos: es una red social dónde el contenido es público y accesible. Además, como comentaremos más adelante, ofrecen una API sencilla y amigable de utilizar. Por hablar de las desventajas de twitter, el hecho de estar limitado a 140 caracteres nos puede perjudicar pues puede haber mucho ruido al tratar los datos.

12.2 API de Twitter

Twitter pone a nuestra disposición una API para leer y escribir datos en Twitter. Podemos hacer búsquedas, crear nuevos tweets, recuperar los tweets de un usuarios, etc...

Existen dos APIs: la REST API y la Streaming API. La Streaming API se usa si queremos obtener información en tiempo real, la REST API es mejor para realizar búsquedas por lo que nos decantamos por ésta. Debemos tener una cuenta asociada para poder utilizarla.

Tenemos ciertas restricciones por parte de Twitter con respecto al numero de Tweets que podemos buscar. Además hay una ventana de 15 minutos para volver a realizar peticiones una vez llegado al limite.

[11] http://www.bbc.co.uk/news/business-12889048

Debemos autenticar nuestra aplicación con los credenciales OAuth que nos proporciona Twitter para tener acceso a la API. Nosotros solo vamos a utilizar la opción de busqueda, tenemos la posibilidad de buscar los tweets más recientes, los más relevantes (con más retweets) o una mezcla de ambos.

12.3 Recolección de datos

En un primer momento realicé un script que, utilizando la API de Twitter, hacia búsquedas en todos los idiomas y guardaba los resultados en un fichero de texto. Esto es porque se puede buscar en todos los idiomas o en uno solo, aunque inicialmente nos interesaba el inglés, español y francés. El script estaba continuamente ejecutándose y cuando llegaba al limite de búsquedas se queda dormido por 15 minutos y luego volvía a buscar.

En total, se han recogido alrededor de 5 millones de tweets en distintios idiomas. Ocupan unos 500 Mb. Hemos decidido separar en 3 idiomas, inglés, francés y español y el resto ignorarlo aunque el fichero lo conservaremos para el futuro.

A continuación la tabla con la cantidad de tweets en los idiomas una vez eliminados los duplicados:

Inglés	Francés	Español
840.000	120.000	55.000

El principal problema es que el inglés domina demasiado por lo que la mayoría de tweets que recogíamos eran en inglés. Aun así, pensamos que con esta cantidad de tweets podemos trabajar y es suficientemente significativa por lo que paramos la recolección de datos en este punto.

13 Filtrado

13.1 Limpieza de datos

Antes de empezar a tratar con los datos recolectados debemos realizar un limpiado de manera que eliminemos aquello que es superfluo y no nos aporte nada. También vamos a separar los tweets según el idioma para tratarlos separadamente. Utilizaremos para ello la librería de python *langdetect*[12].

Vamos a realizar los siguientes tratamientos a los tweets recolectados:

- Eliminación de duplicados
- Eliminación de URLs
- Eliminación de nombres de usuario (cuando se hace referencia a un usuario)
- Transformación a minúsculas
- Separación por idioma (inglés, español y francés)

Más adelante se harán otros tratamientos de los datos pero que no se guardan como eliminación de stopwords o stemmización.

13.2 Filtrado de tweets de "Orange"

Queremos obtener los tweets que se refieran a la empresa Orange y por tanto hay que separarlos del resto de tweets que contienen la palabra "orange" pero no se refieren a ésta. Para conseguirlo vamos a atacar el problema de dos formas: la primera realizar un filtro con palabras clave tales como "Internet","sms" o "tarifa". Esto nos permite en un primer momento obtener de forma sencilla. La segunda será mediante aprendizaje automático, tendremos un modelo al cual le pasamos tweets que hablan de la empresa y tweets que tienen la palabra "orange" pero no hablan de la empresa.

[12]https://pypi.python.org/pypi/langdetect

13.3 Primer método: filtrado por palabras

Para este tipo de filtro he creado una lista de palabras que están relacionadas con la empresa "Orange" tanto en español,francés e inglés. Después simplemente he de coger los datos correspondientes a un idioma, y pasarle este script para que busque en cada tweet si encuentra alguna de las palabras del filtro. Utilizamos stemmización para este proceso, en concreto el SnowBall Stemmer de NLTK. Más adelante compararemos los resultados con los de aprendizaje automático.

13.4 Segundo método: aprendizaje supervisado

Para el segundo método escogeremos un algoritmo de aprendizaje supervisado, y crearemos un modelo que contendrá tweets relacionados con la empresa y tweets con la palabra "orange" no relacionados con la empresa.

Primeramente debemos empezar a manualmente separar tweets que hablan de la empresa de los que no hablan. Estos serán usados para crear el modelo que luego utilizaremos para clasificar tweets.

Una vez tenemos etiquetados los datos, vamos a separar un 20% que serán datos de prueba y el resto será de entrenamiento. Los datos de entrenamiento son aquellos datos que usará el algoritmo para aprender y crear el modelo y los de prueba, son los datos en los que el algoritmo va a intentar predecir a que grupo pertenece.

Utilizamos scikit-learn y hemos escogido el algoritmo SVM dado que en pruebas anteriores con datos más sencillos daba mejores resultados que Multinomial Naive Bayes. Transformamos los datos a la matriz TF-IDF y eliminamos tanto los stopwords (palabras que no aportan información y son frecuentes en el lenguaje, tales como *de,hasta,la,haber*) como términos demasiado frecuentes (que están presentes en más del 85% de los tweets por lo que no son determinantes) y términos que son muy poco frecuentes, ya que no nos aportan nada y seguramente sean palabras mal escritas. Además, eliminamos símbolos como o # y signos de puntuación.

13.5 Resultados

A continuación mostraremos los resultados obtenidos. Para el aprendizaje del algoritmo hemos utilizado como datos de muestra 10.000 tweets. Hemos realizado cross-validation para asegurarnos que los resultados son lo más precisos posibles:

	Aprendizaje supervisado	Filtro palabras
Accuracy	92.4 %	81 %

Vemos que el aprendizaje supervisado es superior al filtro con palabras. Esto es debido a que hay una clara distinción entre los tweets que hablan de la empresa "Orange" y los que hablan de otras cosas. Por ejemplo, se habla mucho de una serie de televisión llamada "Orange is the new black" por lo que el algoritmo aprende a distinguir fácilmente los que hablan de la serie. O en el caso del francés, cuando se habla de "jus d'orange" (zumo de naranja), el vocabulario es muy diferente entre los que hablan de la empresa y los que hablan de otros cosas, por lo que aquí el aprendizaje supervisado es ideal.

El filtrado de palabras es demasiado ingenuo y aunque no tiene mal porcentaje de acierto, claramente es inferior al otro método.

14 Análisis de sentimiento

El análisis de sentimiento intenta detectar la actitud de un interlocutor o escritor con respecto a un tema o la polaridad general de un documento. Una tarea básica en análisis de sentimientos es determinar si la opinión expresada en un documento es positiva o negativa. También se puede ir más lejos y detectar estados emocionales como "triste" o "enfadado" pero nosotros nos centraremos en lo básico del análisis de sentimiento.

Vamos a atacar el problema también de dos formas, una con aprendizaje automático y otra mediante un diccionario de palabras con sentimiento.

14.1 Análisis de sentimiento mediante aprendizaje automático

Para los datos de entrenamiento podríamos manualmente ponernos a clasificar tweets positivos y negativos y con ellos entrenar el algoritmo. El problema es que se necesitan una gran cantidad de ellos y tomaría mucho tiempo. Hay una forma más sencilla de hacerlo y es la siguiente: para recuperar tweets positivos simplemente buscamos tweets que contengan el emoticono ":)" . De esta manera, obtendremos tweets con vocabulario positivo que podremos usar para entrenar nuestro modelo. Hacemos lo mismo para los tweets negativos pero con ":(" . Esta idea esta sacada del paper *Twitter Sentiment Classification using Distant Supervision*[19].

14.2 Análisis de sentimiento con diccionario

Para realizar el análisis de sentimiento utilizando un diccionario debemos crear una lista de palabras positivas y otra de palabras negativas. Hay por internet lexicones de diferentes idiomas en los que se les da a las palabras un valor de sentimiento. Desgraciadamente la mayoría son para el idioma inglés así que para el español y francés hay que improvisar un poco, aunque finalmente se ha logrado crear un lexicón sencillo pero que nos vale basándome en el de Agustín Gravano[13].

[13]http://habla.dc.uba.ar/gravano/sdal

La manera más sencilla de enfocarlo es la siguiente: cogemos un tweet y si encontramos una palabra de nuestro diccionario de términos catalogados como positivos, sumamos uno al valor total del tweet. Si es negativo restamos uno. Al final, si el valor total del tweet es positivo decimos que el tweet es positivo, negativo si es negativo y neutral si es cero.

Este método sencillo encuentra un inconveniente que vamos a ilustrar con un ejemplo:

Me parece que la ultima película de Star Wars no está ***mal****.*

Nuestro algoritmo encontraría la palabra "mal" y etiquetaría el tweet como que está hablando negativamente. En realidad, este tweet debería tener un sentimiento más bien positivo.

Para solucionarlo vamos a aplicar una técnica que consiste en tener una lista de palabras que cambian la polaridad de la siguiente palabra, como por ejemplo "no","nada" o "nunca" en el caso del español o "not","isn't" o "can't" en inglés. Al encontrarnos con una de estas palabras, lo que hacemos es que la siguiente palabra que se encuentre en nuestro diccionario tendrá el signo cambiado, es decir si era positiva ahora será negativa y viceversa. De esta manera afinamos nuestro algoritmo.

14.3 Resultados

A continuación mostramos los resultados. El análisis de sentimiento se ha hecho sobre 500 tweets etiquetados por un humano.

	SVM	Naïve Bayes	Diccionario
Accuracy	65.8 %	71.7 %	69.5 %

Vemos que Naive Bayes es el que mayor porcentaje ha obtenido. Sin embargo, no está lejos del método que utiliza un diccionario y podríamos pensar que mejorando el diccionario un poco estarían igual. Por tanto, no podemos decir que un método sea mejor que otro.

En el apartado final del proyecto analizaremos más en profundidad estos resultados.

15 Extrayendo información

Nos interesa realizar minería de datos a los tweets que hemos descargado con el fin de ver si podemos descubrir algún tipo de información. Ya hemos hablado sobre el clustering al inicio de este documento. Vamos a aplicar algoritmos que intentarán descubrir patrones en nuestros datos. Además del clustering aplicaremos otros métodos que nos permiten obtener información sobre nuestros datos.

Pero,¿cómo podemos saber que tan bueno es un cluster? No es algo como la clasificación en la que puedes saber la precisión de tu algoritmo, aquí se forman grupos, es más difícil saber que tan bueno es el algoritmo.

15.1 Evaluación de clusters

Ya hemos dicho que es difícil saber qué tan bueno son los clusters que se han formado, esto es porque el algoritmo asigna a un cluster documentos que el considera similares, bajo un criterio. No es fácil saber si un documento debería pertenecer a tal o cual cluster o incluso si los clusters que se han formado tienen sentido.

Para evaluar los métodos de clustering tenemos dos opciones: utilizando datos ya etiquetados que no se han usado para el clustering (evaluación externa) o si no disponemos de datos etiquetados, usamos el propio modelo para evaluar (evaluación interna). Nosotros aquí el objetivo con el que vamos a utilizar el clustering es descubrir información por lo que tener datos etiquetados no tiene sentido. Usaremos pues un método que no se basa en datos etiquetados.

Los métodos que se basan en la evaluación interna típicamente dan una puntuación alta al algoritmo que produce clusters con una alta similitud dentro de los clusters y una baja similitud entre clusters. Este metodo tiene sus desventajas ya que un algoritmo como K-means que optimiza las distancias de objeto, puede sobrevalorar el resultado del agrupamiento. Aun asi, nos puede dar una ligera idea de que tal son los clusters formados.

15.1.1 Coeficiente de silueta

El coeficiente de silueta contrasta la distancia media a elementos en el mismo grupo con la distancia media a elementos en otros grupos. Los objetos con un valor de silueta alto están considerados bien agrupados, los objetos con un valor bajo pueden ser ruido o anomalías. Este método es ideal para K-means, y es frecuente verlo utilizado para determinar el número óptimo de grupos.

El coeficiente de silueta para un objeto **s** viene dado por:

- **a** - La distancia media entre el objeto y el resto de objetos en el cluster
- **b** - La distancia media entre el objeto y todos los objetos en el cluster más cercano

$$s = \frac{b - a}{\max(a, b)}$$

15.2 Clustering

Vamos a aplicar clustering a nuestro base de datos de tweets para ver que tipo de información podemos obtener.

Aplicaremos el algoritmo K-means que hemos explicado aquí. Se han realizado pruebas con otros algoritmos como por ejemplo DBSCAN, éste algoritmo formaba demasiados clusters sin sentido,los resultados fueron muy pobres, no pensamos que sea adecuado para Twitter.

Para ver como se han formado los clusters, veremos los términos más frecuentes de cada cluster para hacernos una idea de que tema son los clusters.

Una primera idea, es ver si podemos separar los tweets que hablan de la empresa Orange de los que no hablan. Vamos a ejecutar el K-means para que nos forme dos clusters, osea K = 2:

- Cluster 0 : orange, hola, si, gracias, vodafone, internet, datos, servicio, movistar, ver
- Cluster 1 : the, new, black, is, orange, temporada, ver, serie, viendo, quiero

El algoritmo ha asignado cada tweet a un cluster. Luego vemos aquí los términos mas frecuentes entre los tweets asignados a cada cluster, así podemos hacernos una idea de qué hablan.

A simple vista parece que nos ha formado un cluster que habla de la compañía, esto lo sabemos porque se menciona "vodafone","internet" y "datos", y otro en el que se menciona la serie de TV "Orange is the new black".

Vamos a realizar otra ejecución del algoritmo pero esta vez con K = 3, es decir 3 clusters y quitando los tweets que hablan de "Orange is the new black":

Cluster 0	va, mierda, si, internet, ser, bien, ver, puta, mal, wifi
Cluster 1	multa, millones, 350, francia, posición, dominante, récord, abuso, euros
Cluster 2	vodafone, movistar, jazztel, yoigo, así, fibra, precios, encargado, fútbol

Si analizamos un poco los términos, vemos que el Cluster 0 se refiere al estado de la conexión a internet o wifi de los usuarios, vemos palabras como "va","mal","bien" o descalificativos como "mierda" y "puta". Podemos deducir que hay gente que se está quejando de que le va mal internet, si miramos algunos tweets de este cluster vemos que así es:

> *bufff ha sido cambiar la adsl a orange y todo va mucho más lento. más de 5 minutos pata bajar una app de menos de 3mb. penoso*

> *¡urgente! despues de un año pidiendo a orange que me haga unas facturas bien, ayer me llegaron de nuevo mal.*

El Cluster 1 parece que habla de una multa record de 350 millones a Orange por abuso de posición dominante. En efecto, hay muchos tweets que hablan de ello:

> *multa récord de 350 millones a orange por su posición dominante en francia*

> *francia pone una multa récord a orange por práctica anticompetitiva*

El Cluster 2 parece que habla de otras compañías telefonicas y además contiene las palabras "fibra" y "precio". Parece indicar que comparan a las grandes compañías:

> *recordemos que mediapro tiene grandes competiciones, y vodafone y orange los derechos para bares*

> *a pesar de los datos negativos, movistar mejora, vodafone se mantiene y orange y yoigo caen*

15.2.1 Conclusiones

Bien, hemos visto que se puede extraer información sobre los tweets utilizando algoritmos de clustering. Viendo los términos más frecuentas hemos logrado obtener cierta información sobre lo que se habla en twitter al respecto de la empresa.

No obstante, hay que aclarar que los cluster formados no tienen mucha consistencia. Si analizamos los tweets asignados a los clusters de los ejemplos anteriores, muchas veces hay tweets que no guardan relación con el resto del cluster. Esto es porque los tweets contienen mucho ruido, palabras mal escritas o información superflua que no aporta nada. Es por ello que aunque aquí gracias a los términos más frecuentes obtengamos información referente al cluster, si vamos dentro del cluster, encontraremos tweets que no tienen nada que ver con esa información.

En resumen, si bien hacer clustering nos pueda dar información sobre lo que se dice, los clusters formados no muy buenos, de hecho el coeficiente silueta suele ser muy bajo. Los tweets son textos demasiado cortos y con demasiado ruido para formar cluster coherentes.

15.3 Latent Dirichlet Allocation

LDA[14] es un modelo generativo que permite que conjuntos de observaciones puedan ser explicados por grupos no observados que explicarían por qué algunos datos son similares. Dado un documento, presuponemos que es una mezcla de categorías (o temas) y que las palabras del documento se deben

[14] https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation

a una categoría a la que pertenecería el documento. En resumen, se trata de un manera de automáticamente descubrir categorías o temas dados una colección de documentos.

Una explicación más detallada[15] seria decir que LDA representa los documentos como una mezcla de categorías, que genera palabras con cierta probabilidad. Asume que a la hora de crear los documentos, éstos se han generado de la siguiente manera:

- Decidir el numero de palabras N que tendrá el documento
- Escoger una mezcla de categorías a las que el documento pertenecerá, de acuerdo a una distribución Dirichlet de entre K categorías fijadas (por ejemplo 1/3 comida y 2/3 animales)
- Generar cada palabra del documento primero de la siguiente manera:
  - Seleccionando una categoría de acuerdo a la distribución del anterior apartado
  - De acuerdo a la distribución multinomial de la categoría, generar la palabra. Por ejemplo si escogimos la categoría de comida, generaremos la palabra "pizza" con un 25% de probabilidad, 10% sopa, etc.

Asumiendo este modelo para una colección de documentos, LDA intenta volver hacia atrás para descubrir las categorías que generaron los documentos.

15.3.1 Aprendizaje

Ahora supongamos que tenemos una colección de documentos y hemos fijado un numero K de categorias que queremos descubrir. El algoritmo nos dirá la representación de cada documento y las palabras asociadas a cada categoria:

- Asignar aleatoriamente a cada palabra una categoría
- Por cada palabra W en el documento D, asumimos que todas las asignaciones son correctas excepto ésta. Con nuestro modelo, actualizamos

[15] http://blog.echen.me/2011/08/22/introduction-to-latent-dirichlet-allocation/

la asignación actual de la palabra comprobando la asignación en otros documentos para es palabra, y la asignación de categorías en el documento.

Después de ejecutar el anterior varias veces tendremos un estado en el que las asignaciones son bastante buenas.

15.3.2 Aplicando LDA a Twitter

Vamos a ver que tal se comporta LDA con nuestros datos de Twitter.

Topic 0	orange gracias si fibra móvil servicio hacer banco 10 casa
Topic 1	orange vodafone movistar voy liga va bien solo tv bouygues
Topic 2	hola si datos internet puedo dm buenas nuevo puedes vez
Topic 3	orange the new black is ver temporada ahora ser quiero

Parece ser que LDA nos ha creado categorías algo más confusas que cuando utilizamos K-means. Aparte de la categoría de la serie de TV, el resto no tiene mucho sentido. Quizás el Topic 2 se refiera a consultas que hacen al Twitter de Orange ya que vemos "hola","buenas" y "puedes" ,"datos", parece que le piden que le faciliten los datos para ayudarle a resolver el problema.

El Topic 1 podría hablar sobre los partidos de la liga en TV y que operadoras lo ofrecen, al parecer están discutiendo sobre los derechos de la liga de fútbol actualmente.El Topic 0 es demasiado genérico y no nos dice gran cosa.

En general, incluso probando con temas distintos a "Orange", LDA parece que no crea categorías demasiado buenas. Quizás habría que realizar más limpieza y filtrado de los tweets pero parece que no nos aporta más que el clustering.

16 Conclusiones

El objetivo de este trabajo era ver la posible aplicación de los algoritmos de clustering y de aprendizaje automático con Twitter. A lo largo de este proyecto, se han estudiado distintos algoritmos así como herramientas relacionadas con data mining. Era un tema que no había tocado mucho, de hecho algoritmos de aprendizaje automático no había más que visto brevemente en clase la teoría.

En lo que respecta a la red social Twitter, nos ofrece una API fácil de utilizar pero que viene limitada. Hubiese sido más fácil trabajar con un acceso que no estuviese tan limitado pero no ha supuesto grandes inconvenientes. Se ha visto que los tweets están lleno de ruido, la gente escribe con muchas faltas de ortografía o expresiones vulgares y mucho spam. Al tener un limite de 140 caracteres, estamos un poco limitados a la hora de trabajar con algoritmos que hacen uso de procesamiento de lenguaje natural o análisis de texto.

En cuanto a filtrar mediante aprendizaje supervisado los tweets que hablan de la empresa Orange o que no, es un método muy efectivo ya que las palabras que hablan de uno u otro son bien distintas por lo que los algoritmos lo van a diferenciar con facilidad. Es perfectamente factible realizar esta operación automáticamente, llegamos a tener un 92% de precisión.

El análisis de sentimiento de los tweets tiene un porcentaje de precisión similar entre usar aprendizaje supervisado o un algoritmo usando diccionario. Los porcentajes obtenidos, alrededor del 70%, parecen buenos ya que según estudios[16] incluso los humanos no estarían de acuerdo el 100% al dar una respuesta. Se calcula que los humanos están de acuerdo un 79% de las veces. Investigando sobre estudios de análisis de sentimiento, ambas técnicas por diccionario y aprendizaje, son utilizadas y perfectamente válidas.

Un problema que nos encontramos al hacer análisis de sentimiento es como detectar la ironía. "Qué buena película." puede ser un comentario irónico y que en realidad esté hablando mal de la película. Es muy difícil detectar esto por lo que hay que tener en cuenta esta limitación a la hora de

[16] http://mashable.com/2010/04/19/sentiment-analysis/#SbfOTkL_Aiqd

analizar tweets. Un dato curioso que he visto al hacer análisis de sentimiento es que la gente habla mal de algo tres veces más que habla bien. La gente se queja mucho en Twitter :)

Finalmente, el clustering aplicado a Twitter, nos permite extraer ciertas informaciones para ver de que hablan. Nos hacemos una idea general pero al ser un solo tema (en este caso Orange), es más difícil sacar información relevante. Si fuese un baturrillo de temas seguramente los algoritmos se comportarian mejor y crearían clusters con mejor calidad. Aun así, tanto el clustering como el LDA, son una ayuda para ver por encima los temas de que se hablan en los tweets que tenemos, pero la calidad de los clusters deja mucho que desear. Los algoritmos de clustering son mejores cuando hay más texto, para documentos más largos.

16.1 Trabajo futuro

Como trabajo futuro quedaría seguir recolectando más tweets para aumentar la base de datos, los modelos que hemos creado mediante aprendizaje automático deben irse actualizando ya que sino se quedarán anticuados rápidamente.

Hay que ver como eliminar todo el ruido de los tweets que al final nos impide trabajar con comodidad con los tweets. Mejorar el diccionario, no solo añadiendo mas palabras sino dándoles una puntuación a cada palabra basada en que grado de sentimiento expresan. No tiene el mismo peso "excelente" que "bueno". Podríamos utilizar etiquetado gramatical[17] para realizar un análisis de las palabras más profundo. Además, habría que poder corregir las faltas de ortografía para evitar que nos penalicen y que reconociese la jerga de internet como "lol" (acrónimo de Laughing Out Loud).

[17] https://es.wikipedia.org/wiki/Etiquetado_gramatical

References

[1] *Explosión de los datos*. http://aci.info/2014/07/12/the-data-explosion-in-2014-minute-by-minute-infographic/.

[2] *Minería de datos*. https://en.wikipedia.org/wiki/Data_mining.

[3] *Aprendizaje supervisado*. https://en.wikipedia.org/wiki/Supervised_learning.

[4] *Aprendizaje no supervisado*. https://en.wikipedia.org/wiki/Unsupervised_learning.

[5] *Lemoteur.fr*. http://www.lemoteur.fr/.

[6] *Cuota de mercado navegadores en Francia*. http://www.journaldunet.com/ebusiness/le-net/parts-de-marche-des-moteurs-de-recherche-en-france.shtml.

[7] *Guía oficial ElasticSearch*. https://www.elastic.co/guide/index.html.

[8] *Solr vs ElasticSearch*. http://solr-vs-elasticsearch.com/.

[9] *Clustering*. https://en.wikipedia.org/wiki/Cluster_analysis.

[10] *Guía métodos clustering*. http://scikit-learn.org/stable/modules/clustering.html.

[11] *K-means*. https://es.wikipedia.org/wiki/K-means.

[12] *K-means++*. http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf.

[13] *K-nearest neighbors*. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm.

[14] *DBSCAN*. https://es.wikipedia.org/wiki/DBSCAN.

[15] *SVM*. http://scikit-learn.org/stable/modules/svm.html.

[16] *Naives Bayes*. http://scikit-learn.org/stable/modules/naive_bayes.html.

[17] *Decision tree*. http://scikit-learn.org/stable/modules/tree.html.

[18] *Cluster validation*. http://www.cs.kent.edu/ jin/DM08/ClusterValidation.pdf.

[19] *Twitter Sentiment Classification using Distant Supervision*. http://cs.stanford.edu/people/alecmgo/papers/TwitterDistantSupervision09.pdf.