Capítulo 3

Metodología

En el capítulo 2 se presentaron algunas generalidades de los sistemas de minería de opiniones. En este capítulo, se describen los métodos usados en este proyecto para la creación del sistema de clasificación de opiniones.

Antes de describir el sistema, es necesario presentar las herramientas de programación utilizadas para su creación.

3.1. Herramientas de programación utilizadas

3.1.1. Python

Se eligió el lenguaje de programación Python[1] debido a que es un lenguaje simple con excelente funcionalidad para procesar información lingüística [24].

Las fortalezas más importantes de Python, según [43], son:

- La librería estándar de Python es suficientemente amplia para que sea considerado apto para la resolución de cualquier tipo de problema informático (desarrollo Web, acceso a bases de datos, aplicaciones de escritorio, desarrollo científico y numérico, entre otros).

[1] http://www.python.org/

- Es compatible con otros lenguajes de programación. Se puede integrar con librerías de Java, a través de Jython[2]. También es compatible con .NET (lenguajes como C#, Visual Basic o F#) por medio de IronPython[3]. Incluso puede acoplar módulos desarrollados en C o C++, cuando no se encuentra una librería adecuada o se requiere del desempeño del código de bajo nivel.
- Python puede ser ejecutado en cualquier sistema operativo moderno, por ejemplo Windows, Unix/Linux, OS/2, Mac, Amiga, entre otros.
- Python está implementado bajo una licencia tipo *open source*. Esto hace que sea utilizado y redistribuido libremente.

La versión de Python utilizada fue la 2.6.4.

Existen tres módulos para Python que fueron fundamentales para el desarrollo de este trabajo, a continuación se describen brevemente cada uno de estos módulos:

- NLTK (*Natural Language Toolkit*)[4]: conjunto de librerías enfocadas al PLN. Es ideal para la investigación y desarrollo en esta área y también en las relacionadas como minería de textos, inteligencia artificial y aprendizaje de máquinas. NLTK ofrece clases básicas para representar datos relevantes al PLN, así como interfaces para realizar tareas tales como etiquetado PoS (*Part of speech*), búsqueda de n-gramas, obtención de frecuencias de aparición, entre otras muchas tareas útiles. También cuenta con una amplia documentación y una actividad bastante grande por parte de sus usuarios en foros y en listas de correos.
- Numpy[5]: es un módulo para Python que incluye un gran soporte para arreglos, vectores y matrices multidimensionales así como las funciones matemáticas (del álgebra lineal) para este tipo de objetos. También incluye sus propios tipos de datos, usados en los elementos de las matrices.
- Scipy[6]: Es una librería de algoritmos y herramientas matemáticas para Pyt-

[2]Jython es una implementación de Python para la máquina virtual de Java.
[3]IronPython es una implementación de Python orientado a la tecnología .NET.
[4]http://www.nltk.org
[5]http://numpy.scipy.org
[6]http://www.scipy.org

hon. Contiene el modulo de álgebra lineal, que resulta indispensable para el algoritmo de agrupamiento ya que incluye al objeto que representa a una matriz dispersa, útil para crear y manipular matrices con una gran cantidad de ceros de forma eficiente. Scipy le permite a Python ser un lenguaje muy utilizado en la investigación científica actualmente.

También se usaron los módulos IMDBpy[7] y Beautiful Soup[8]. IMDb (*Internet Movie Database*) es una base de datos en línea con información acerca de películas, series de televisión, actores, equipos de producción, videojuegos, entre otros temas. Permite a cualquier usuario registrado en el sitio escribir una reseña y asignar una calificación de 1 al 10 a cualquiera de los temas mencionados. La calificación general de una película, por ejemplo, es el promedio de todas las calificaciones vertidas por los usuarios sobre esa película. IMDBpy es utilizado para recuperar y administrar la información de IMDb . Sin embargo, al momento de programar el sistema, IMDBpy no ofrecía forma de recuperar los comentarios de cada película. Es por esto que fue necesario programar un Web crawler, cuya explicación está más abajo. Para programar este crawler se necesita el módulo Beautiful Soup, el cual realiza la función de analizar código HTML y XML. Tiene la ventaja de poder trabajar con lenguaje de marcado mal formado y provee métodos para navegar, buscar y modificar el árbol de análisis de un documento.

3.1.2. Eclipse IDE (*Integrated Development Environment*)[9]

Eclipse es un ambiente de desarrollo de software conformado por un entorno de desarrollo integrado y un sistema de *plugins* o complementos. Está escrito principalmente en Java. Puede ser usado para desarrollar aplicaciones en Java, por medio de complementos en Ada, C, C++, COBOL, Perl, PHP, Python, Ruby y Scheme, entre otros.

La versión Galileo 2.5 fue la utilizada para la programación en Python.

[7] http://imdbpy.sourceforge.net/
[8] http://www.crummy.com/software/BeautifulSoup/
[9] http://www.eclipse.org

3.1.3. Pydev[10]

Pydev es el complemento que permite programar en Python, Jython e IronPython en el ambiente de desarrollo Eclipse. Ofrece refactorización de código, depuración gráfica y análisis de código automático, entre otras útiles funciones. La versión utilizada fue la 1.6.4. Una vez instaladas y configuradas las tres herramientas, se comenzó con la programación del sistema.

3.2. Procesos del sistema

En esta sección se describen los procesos que comprenden al sistema. Así, en la figura 3.1 se pueden apreciar los procesos principales.

Dentro del sistema se pueden identificar ocho procesos principales, con sus respectivos datos o archivos de entrada y de salida. A continuación se detallará cada uno de ellos.

3.2.1. Obtención de artículos sobre películas desde Wikipedia[11]

Por medio de la opción Special:Export[12] que ofrece Wikipedia, se pueden exportar los artículos que cumplan con cierta categoría elegida por el usuario. De esta forma se extrajeron dos archivos XML, uno para películas de 2009 y otro para películas de 2010 (*2009 films* y *2010 films*, respectivamente).

Un fragmento del contenido de este tipo de archivos aparece en la figura 3.2. Estos archivos están bien formados de acuerdo a los lineamientos de XML. Sin embargo, contienen un lenguaje de marcado propio de Wikipedia, el cual hay que analizar para poder extraer los nombres y fechas. Las fechas son necesarias ya que, para limitar la cantidad de información, se usaron solo las películas estrenadas en noviembre y diciembre de 2009 y las estrenadas en enero de 2010.

[10] http://pydev.org/

[11] Enciclopedia libre, en línea y poliglota de la fundación Wikimedia. Cuenta con más de 3.5 millones de artículos en inglés, mismo idioma de las reseñas y opiniones sobre películas usadas en esta tesis.

[12] http://en.wikipedia.org/wiki/Special:Export

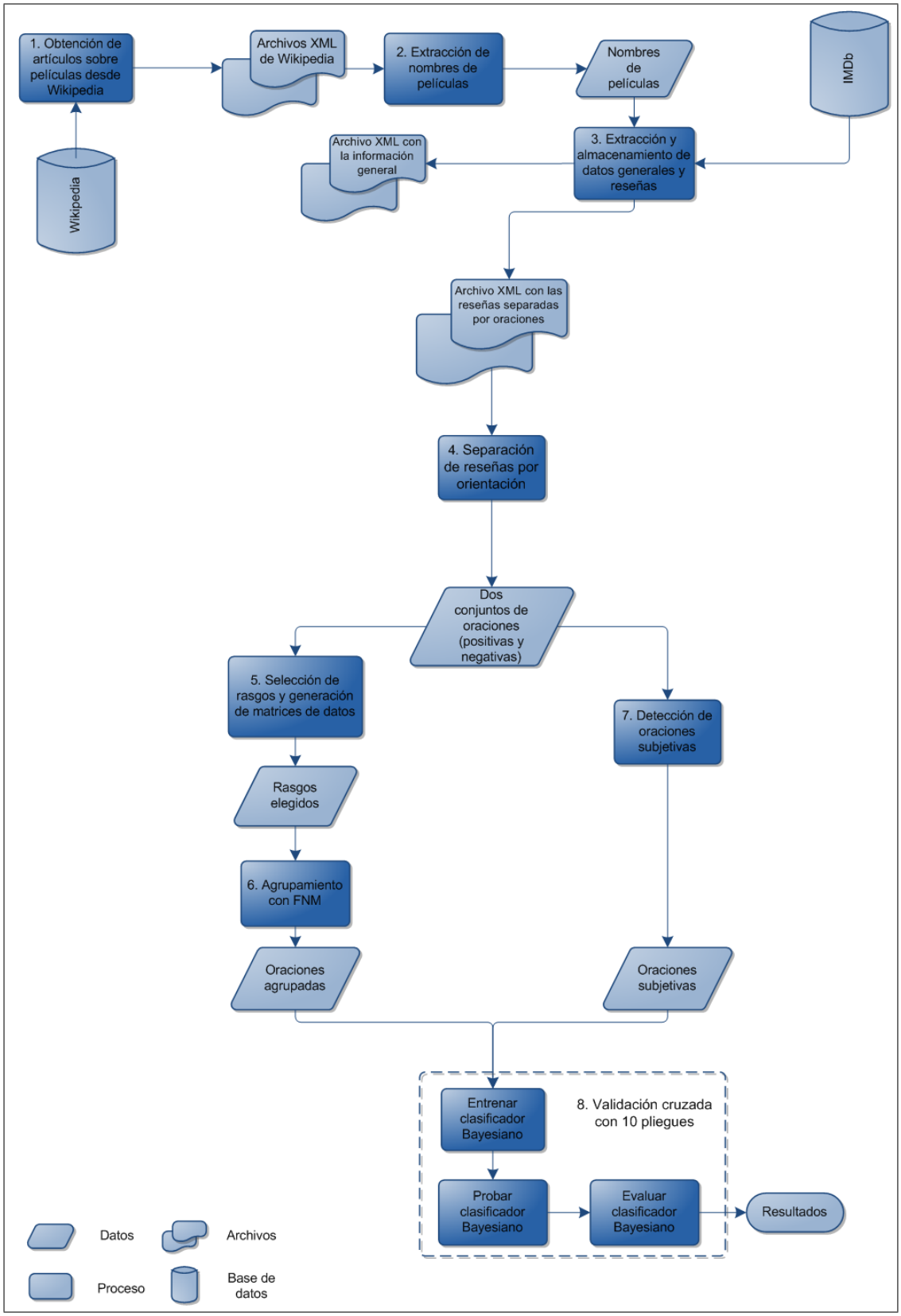

Figura 3.1: Procesos que comprenden al sistema de clasificación - La figura muestra los procesos que se realizaron para clasificar los enunciados dependiendo de su opinión

```
[[Category:2009  in India]]
[[Category:2009 films | ]]
[[Category:Lists of 2009 films by country or language|Tamil]]</text>
    </revision>
  </page>
  <page>
    <title>Category:Lists of 2009 films by country or language</title>
    <id>23835047</id>
    <revision>
      <id>306315314</id>
      <timestamp>2009-08-06T01:11:30Z</timestamp>
      <contributor>
        <username>Rich Farmbrough</username>
        <id>82835</id>
      </contributor>
      <text xml:space="preserve">{{Films by country or language|2009}}</text>
    </revision>
  </page>
  <page>
    <title>(500) Days of Summer</title>
    <id>18057739</id>
    <revision>
      <id>341125687</id>
      <timestamp>2010-01-31T18:43:13Z</timestamp>
      <contributor>
        <ip>68.4.102.67</ip>
      </contributor>
      <comment>/* Home media */</comment>
      <text xml:space="preserve">{{Infobox film
| name           = (500) Days of Summer
| image          = Five hundred days of summer.jpg
| caption        = Promotional film poster
| director       = [[Marc Webb]]
| producer       = Mason Novick &lt;br /&gt;Jessica Tuchinsky &lt;br /&gt;[[Mark Waters (director)|Mark Waters]]
&lt;br /&gt;Steven J. Wolfe
| writer         = Scott Neustadter &lt;br /&gt;Michael H. Weber
| narrator       = [[Richard McGonagle]]
| starring       = [[Joseph Gordon-Levitt]] &lt;br /&gt;[[Zooey Deschanel]] &lt;!-- top listed only (see Poster).
see Cast for more details. --&gt;
| music          = [[Mychael Danna]] &lt;br /&gt;Rob Simonsen
```

Figura 3.2: Archivo `2009_films.xml` - La figura muestra un fragmento del contenido de uno de los archivos XML extraídos desde Wikipedia.

La información buscada está contenida en estructuras llamadas *infoboxes* (o cajas de información) donde aparecen organizados los datos de cada película. Sin embargo, existen ocasiones en las que los datos no están completos, contienen caracteres erróneos o cambian su formato de una película a otra. Estos problemas son solucionados en gran medida durante la siguiente etapa.

3.2.2. Extracción de los títulos de las películas

Con los archivos de Wikipedia obtenidos, se procedió a extraer los títulos o nombres de las películas que en ellos aparecen. Esta tarea se hizo con ayuda de expresiones regulares. Se extrajo el título o nombre de la película y su fecha de estreno (por ejemplo, del archivo que aparece en la figura 3.2, se extrajo el nombre de la película *(500) Days of Summer*). Como se dijo anteriormente, la fecha sirvió para limitar la cantidad de títulos de películas (y por ende de películas) a procesar.

Así, se filtraron los nombres de películas de acuerdo a su fecha de estreno. Los nombres de las que se encontraban dentro del rango usado (noviembre-diciembre 2009 y enero 2010) fueron almacenados en un objeto tipo lista para después buscar su información en la siguiente etapa.

En ciertas ocasiones los meses de las fechas de estreno aparecen con letras (January 2010) y en otras ocasiones, con número (09 - 2009). Esto obligó a generar expresiones regulares que atiendan cada uno de estos casos.

También, algunos títulos contenían caracteres especiales[13], los cuales fueron removidos para evitar problemas al almacenar estos títulos en nuevos archivos XML.

3.2.3. Extracción y almacenamiento de datos generales y reseñas

Con los nombres de las películas se buscaron y extrajeron los datos adicionales y las reseñas de cada película. Como ya se dijo, esta etapa fue realizada con la

[13]En http://meta.wikimedia.org/wiki/Help:Special_characters hay una lista con estos caracteres especiales.

información contenida en el sitio IMDb. De esta manera, se buscó cada nombre de película en IMDb, se extrajo su identificador único dentro de la base y a partir de este se adquirió la siguiente información:

- Titulo en inglés[14].
- Títulos en otros idiomas, en esta base conocidos como *aliases*.
- Calificación o *rating*. Su rango es de 1 a 10 y es la calificación[15] que le dieron los usuarios por medio de sus votos a cada película.
- Número de votos.
- El identificador único, el cual es un número de siete cifras o más.

Después, con el identificador único, se procedió a extraer los comentarios desde las páginas Web que los contienen por medio del Web crawler. Este se encarga de recorrer página por página y recuperar cada uno de los comentarios. Al momento de desarrollar el sistema, las direcciones de las páginas de comentarios tenían la forma:

http://www.imdb.com/title/tt`XXXXXXX`/usercomments?filter=best;start= donde `XXXXXXX` es el identificador único de la película.

El crawler se encargó de obtener el número de páginas de comentarios, de descargar cada página, analizar el código HTML y obtener la información deseada de cada comentario. La información obtenida es la siguiente:

- Utilidad: cada reseña puede ser calificada por otros usuarios, quienes determinan si es útil o no.
- Título del comentario.

[14] Algunas veces fue necesario buscar el título de la película por medio de la fecha de estreno.

[15] Según http://www.imdb.com/help/show_leaf?ratingsexplanation, el promedio está ponderado (*weighted*) con el fin de reducir o eliminar los votos de individuos cuyo fin es cambiar el rating actual de la película en lugar de dar una opinión sincera acerca de ella. El procedimiento para obtener este promedio no es revelado.

- Rating: calificación que el usuario le dio a la película; en un rango de 1 a 10; expresado en el número de estrellas marcadas de diez posibles.

En la figura 3.3 se muestran en su forma original los datos extraídos.

Después de extraer los datos y reseñas, se procedió a almacenarlos de una forma estándar para poder ser ocupados después por el clasificador u otro sistema. Así, ya no es necesario realizar todo el proceso de consultar IMDb, ahorrando tiempo y recursos. Las reseñas se dividieron en oraciones por medio del segmen-

Figura 3.3: Forma en la que se encuentran los datos en la página original de IMDb - La figura muestra una parte de la información, como aparece en el sitio, que se extrajo con el crawler.

tador de oraciones Punkt, que ofrece[16] NLTK. También se eliminaron caracteres especiales. Esto para poder analizarlas y alimentarlas a los algoritmos de clasificación y agrupamiento.

Se generaron dos archivos XML. Un archivo para los datos de cada película, `peliculas.xml` y otro exclusivamente con las reseñas de cada película, `reseñas.xml`.

En el archivo XML de los datos de cada película, `peliculas.xml`, existe el elemento padre `movies`, el cual solo tiene un elemento hijo, `title`, cuyo contenido es el nombre de la película, en inglés. Este elemento `title` tiene tres atributos:

[16]El cual se detalla en: http://nltk.googlecode.com/svn/trunk/doc/api/nltk.tokenize.punkt.PunktSentenceTokenizer-class.html

1. `IMDBid`: el identificador único de la película.

2. `ratingG`: el rating promediado de la película.

3. `votes`: el número de votos que tiene la película.

En la figura 3.4 se observa un fragmento del archivo `peliculas.xml` (datos de cada película).

```
<?xml version="1.0" encoding="UTF-8"?>
 <movies>
    <title IMDbID="1333093" ratingG="3.9" votes="15">Love Me Again (Land Down Under)</title>
    <title IMDbID="0235158" ratingG="7.5" votes="16">Aanaval Mothiram</title>
    <title IMDbID="0233469" ratingG="5.2" votes="23172">Collateral Damage</title>
    <title IMDbID="1343046" ratingG="6.7" votes="97">Meitantei Conan: Shikkoku no chaser</title>
    <title IMDbID="1292655" ratingG="4.8" votes="75">The River Within</title>
    <title IMDbID="1571728" ratingG="NA" votes="NA">First Ink</title>
    <title IMDbID="0289226" ratingG="8.3" votes="18">House Full</title>
    <title IMDbID="1202230" ratingG="7.6" votes="23">Courting Condi</title>
    <title IMDbID="1392996" ratingG="7.5" votes="31">Sarah's Choice</title>
    <title IMDbID="0448182" ratingG="6.0" votes="119">Yesterday Was a Lie</title>
    <title IMDbID="1459219" ratingG="7.0" votes="14">Actresses</title>
    <title IMDbID="1397502" ratingG="8.0" votes="21">Grown Up Movie Star</title>
    <title IMDbID="1059925" ratingG="6.5" votes="1394">Greta</title>
    <title IMDbID="1341167" ratingG="9.0" votes="62">Four Lions</title>
    <title IMDbID="0115195" ratingG="6.9" votes="2407">Gulliver's Travels</title>
    <title IMDbID="1488591" ratingG="7.6" votes="10">Reiton kyôju to eien no utahime</title>
    <title IMDbID="1345777" ratingG="7.9" votes="843">Ishqiya</title>
    <title IMDbID="0085776" ratingG="8.0" votes="201">Katha</title>
    <title IMDbID="0860906" ratingG="8.7" votes="217">Evangerion shin gekijôban: Ha</title>
    <title IMDbID="1066327" ratingG="6.1" votes="17">London Betty</title>
    <title IMDbID="1148165" ratingG="6.9" votes="287">Bran Nue Dae</title>
    <title IMDbID="0120255" ratingG="7.8" votes="15807">The Sweet Hereafter</title>
    <title IMDbID="0878804" ratingG="7.7" votes="25484">The Blind Side</title>
    <title IMDbID="0367942" ratingG="7.3" votes="10">Kutty</title>
```

Figura 3.4: Archivo `peliculas.xml` - La figura muestra un fragmento del archivo XML que contiene la información de cada película. Cuando no se pudo obtener una calificación y/o el número de votos (porque IMDb aún no contaba con esos datos), el valor fue ocupado por las letras "NA" (*Not Available*).

En el archivo XML con las reseñas, `reseñas.xml`, el elemento padre es `todas_reviews`, y su elemento hijo es `comment`, cuyo contenido es toda la reseña de un usuario sobre una película. Este elemento `comment` tiene tres atributos:

1. `IMDBid`: el identificador único de la película.

2. `u`: la utilidad de la reseña que se expresa como x/y. Donde x es el número de usuarios que encontraron útil el comentario de y usuarios que lo revisaron.

3. `r`: rating que el autor de la reseña le dio a la película.

Además, comment tiene dos elementos hijos:

1. `titleC`: es el título de la reseña.

2. `s`: que marca los enunciados de la reseña y ocurre tantas veces como enunciados haya en la reseña en cuestión (recordar que las reseñas fueron segmentadas en oraciones en la sección 3.2.3).

En la imagen 3.5 se observa un fragmento de `reseñas.xml` (con las reseñas de cada película).

Figura 3.5: Archivo `reseñas.xml` - La figura muestra un fragmento del archivo XML que contiene las reseñas de cada película.

Al final, se recuperaron 171 películas y 7,089 reseñas. Las reseñas contaron con 123,878 enunciados en total.

El archivo `reseñas.xml` es el que se utilizó para realizar los experimentos que se describirán a continuación.

En la siguiente etapa, los elementos <s/> (enunciados de los comentarios) se separaron por orientación positiva o negativa.

3.2.4. Separación de reseñas por orientación

En esta etapa se separaron los enunciados de las reseñas en dos conjuntos diferentes. La separación se hizo de acuerdo a la calificación que el comentario dio a la película reseñada. Se consideró que los comentarios que exhibieron un rating (del autor de la reseña) menor a cinco, son de orientación negativa y los que tuvieron un rating igual o mayor a cinco eran de orientación positiva. Solo se trabajó con las primeras 50 reseñas de cada película, para evitar la dominación del corpus por parte de un número pequeño de películas populares.

No se utilizaron los comentarios que no asignaron una calificación a la película. Asimismo, se utilizaron solo las reseñas cuya utilidad fue superior al 50 %. Esto para tratar de eliminar comentarios sin lógica o comentarios basura (anuncios comerciales, letras aleatorias, entre otros).

Los conjuntos de oraciones se almacenaron en memoria para continuar con el proceso del sistema. Las cantidades de oraciones, en los conjuntos de orientación negativa y positiva, son:

- 12,998 enunciados para la orientación negativa.
- 48,323 enunciados para la orientación positiva.

Como se puede apreciar, la cantidad de enunciados positivos es casi cuatro veces mayor a la cantidad de enunciados negativos. Esto se debe simplemente a que existen un mayor número de reseñas positivas que negativas en las películas utilizadas.

Una vez separadas las oraciones, se siguieron dos líneas de experimentación: la agrupación de los comentarios por medio de FNM y la detección de oraciones subjetivas[17]. Después, con las oraciones agrupadas o con las oraciones identificadas como subjetivas, se entrenó un clasificador bayesiano ingenuo.

Primero se explicará el método usado para agrupar por medio de FNM y posteriormente se explicará la identificación de oraciones subjetivas.

3.2.5. Selección de rasgos y generación de matrices de datos

Antes de aplicar el método de agrupación, se generaron las matrices de datos necesarias como entrada para el algoritmo FNM.

Esto es, se crearon dos matrices de datos, una para cada una de las orientaciones (una para la positiva y otra para la negativa). Cada matriz requirió de un vector de rasgos. Estos rasgos pueden ser cada una de las palabras de las oraciones de cada orientación. Sin embargo, para limitar el tamaño del vector de rasgos, se usaron aquellos que, en teoría, ofrecerían mayor información acerca de un comentario positivo o negativo.

Los rasgos que se utilizaron para cada una de las matrices fueron elegidos con base en la experimentación. Los que resultaron más exitosos y, por ende, fueron utilizados se enumeran a continuación:

- Trigramas que aparecen más de 16 veces. Estos trigramas además no deben contener un sustantivo en la segunda posición del trigrama. La razón fue que, con base en la experiencia, los trigramas que contienen sustantivos en esa posición generalmente eran nombres propios, ya sea de películas, actores, directores, etc.
- Adjetivos que aparecen más de cinco veces.
- Palabras (unigramas) que aparecen más de 40 ocasiones y que no se encuentren en una lista de paro.

[17]Como se vio antes, una oración subjetiva es aquella que describe el aprecio o jucio acerca de algo.

Estos rasgos son los que conformaron las columnas de las matrices de datos.

Fue necesario tokenizar las oraciones con el fin de buscar trigramas, etiquetarlos y obtener frecuencias de aparición. Estas tareas se llevaron a cabo con la ayuda del NLTK, que ofrece varios métodos[18] para tokenizar. El utilizado fue el basado en expresiones regulares, que conserva palabras, palabras con guiones, palabras con apóstrofes y números. Estas expresiones fueron construidas con base en las propuestas por [24, 44]. El etiquetado PoS se realizó también con la ayuda del NLTK, usando, como se dijo antes, el conjunto de etiquetas Penn Treebank. Las etiquetas buscadas fueron JJ (adjetivo), JJR (adjetivo comparativo), JJS (adjetivo superlativo) y VBG (adverbio) y NN (sustantivo).

Se generaron entonces las dos matrices, donde los renglones eran los enunciados de las reseñas y las columnas los rasgos seleccionados. El valor de cada celda es la frecuencia absoluta de cada palabra o trigrama en el texto. Se probó también con el peso tf-idf[19], pero los resultados fueron mejores con la frecuencia.

Las matrices creadas fueron del tipo dispersas, ya que la gran mayoría de sus elementos son cero.

Las características de las matrices son:

- La matriz de orientación negativa tuvo una dimensión de $351 \times 12,998$, con 27,878 elementos distintos a cero.
- La matriz de orientación positiva tuvo una dimensión de $1,470 \times 48,323$, con 195,699 elementos distintos a cero.

La figura 3.6 muestra un fragmento de la matriz de orientación negativa representada como una imagen. Esta imagen, en su totalidad, tiene por dimensiones $12,998 \times 351$ pixeles, por lo tanto, cada valor de la matriz está representado por un pixel. Los valores distintos a cero se muestran en color negro.

Con estas matrices se procedió a realizar el agrupamiento FNM.

[18] Estos métodos se pueden revisar en: http://nltk.googlecode.com/svn/trunk/doc/howto/tokenize.html.

[19] El peso tf-idf es una medida estadística usada para determinar la importancia de una palabra en un documento de un corpus. El peso incrementa proporcionalmente a la frecuencia relativa (o absoluta) de la palabra en el documento y disminuye proporcionalmente a la cantidad de veces que aparece esa palabra en otros documentos del corpus.

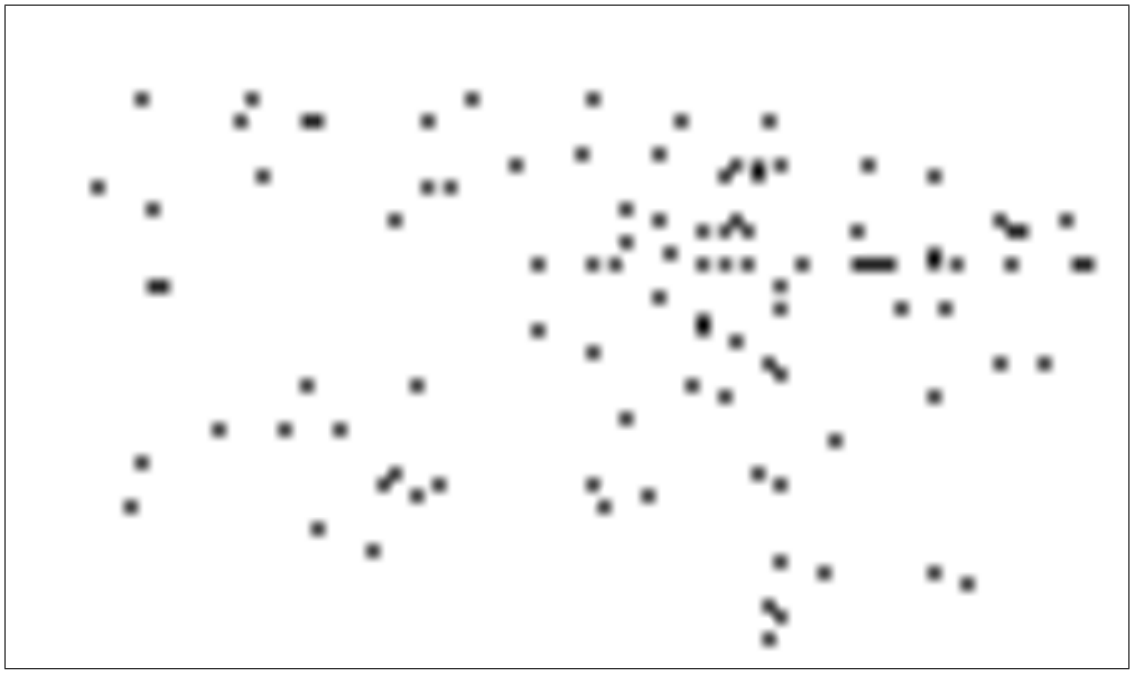

Figura 3.6: Matriz dispersa - Fragmento de una imagen que representa la matriz dispersa de orientación negativa. Los elementos en negro son aquellos mayores a cero.

3.2.6. Agrupamiento con factorización no negativa de matrices (FNM)

El objetivo de agrupar las oraciones es obtener aquellas que contengan las palabras que mejor definen a cada orientación; obteniendo los enunciados más representativos. De esta forma, en los rasgos de las matrices, se cuenta no solo con los unigramas y trigramas más comunes, sino también con las palabras que rodean a estos n-gramas. Con esto se pretende reducir la cantidad de información necesaria para entrenar el clasificador. Otra razón para realizar el agrupamiento es entrenar el clasificador solo con oraciones subjetivas, esto es, dejando de lado las que no ofrecen una opinión.

Como se vio en el capítulo dos sección 2.3.4.2.1, la agrupación por medio de FNM consiste en factorizar la matriz de datos de entrada en dos matrices de menor dimensión. Ambas matrices después se interpretan para obtener los grupos a los cuales pertenece cada oración.

Las matrices de entrada, para cada agrupamiento, fueron: la matriz de orientación negativa y la de orientación positiva, creadas en la etapa anterior. Estas matrices se descompusieron en dos matrices cada una.

Al ejecutar el algoritmo de FNM, es necesario indicar el valor r, que en términos prácticos es el número de grupos que se desean obtener. Se eligió $r = 15$ para los dos procesos de agrupamiento.

Los resultados se almacenaron en memoria para ser procesados en la siguiente etapa. También fueron guardados en archivos de texto plano.

En la figura 3.7 se aprecia un fragmento de los resultados entregados por el agrupamiento:

Cuando se encuentran los grupos, a cada enunciado miembro le corresponde un valor numérico, llamado valor de pertenencia al grupo, el cual es obtenido de las matrices factorizadas. Esta cantidad indica qué tanto pertenece ese enunciado al grupo, por lo que los que tienen un valor mayor, son los más representativos del grupo. Por esta razón, para entrenar el clasificador bayesiano ingenuo, se tomaron los enunciados cuyo valor de pertenencia al grupo fue mayor a la unidad.

Al terminar el proceso de agrupamiento, se obtuvieron las siguientes cantidades de oraciones, cada una con valores de pertenencia superior a uno:

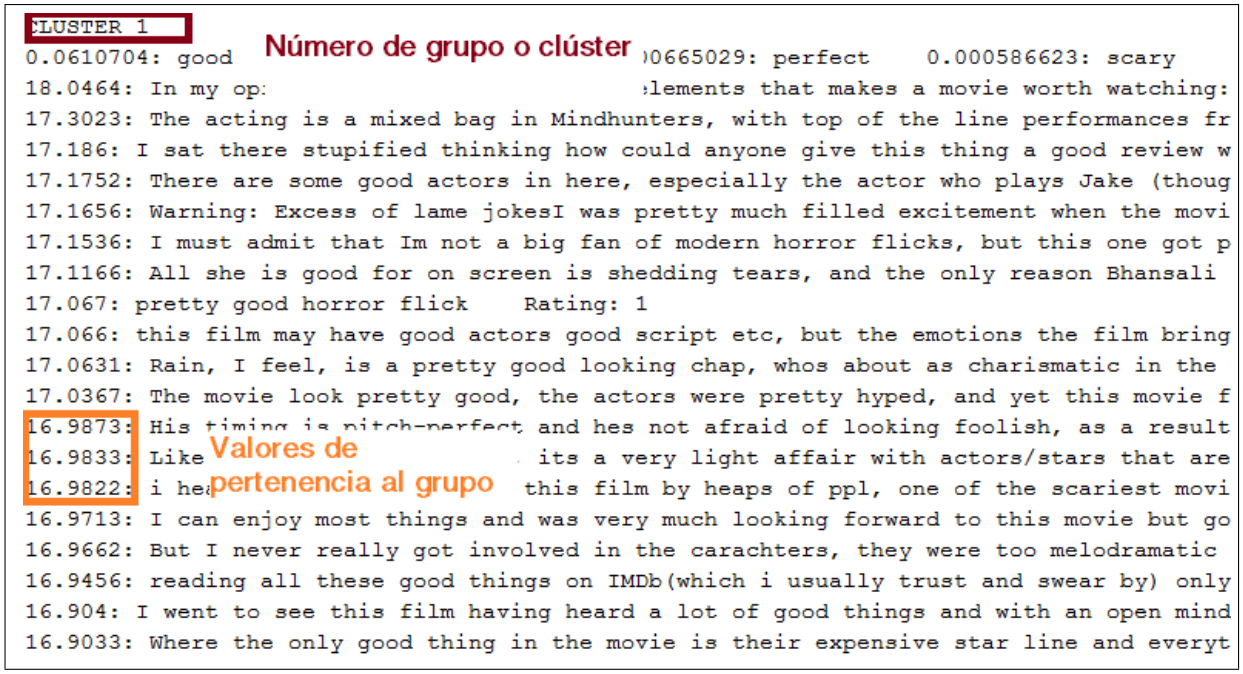

Figura 3.7: Archivo `NMFpositivos.txt` - La figura muestra un fragmento del archivo de texto con el resultado del agrupamiento. Se observan el número del grupo y el valor de pertenencia al grupo de cada enunciado miembro.

- Para la orientación negativa: 5,740 oraciones.
- Para la orientación positiva: 30,058 oraciones.

Con estos enunciados se entrenó el clasificador bayesiano ingenuo, en el marco de la primera línea de experimentación.

La siguiente sección describirá el procedimiento para detectar oraciones subjetivas. Este procedimiento comprende la segunda línea de experimentación.

3.2.7. Detección de oraciones subjetivas

También con los dos conjuntos de enunciados ya separados por calificación como entrada, y como segunda línea de experimentación, se detectaron y utilizaron solo aquellas oraciones subjetivas del texto para entrenar el clasificador.

Se tomaron los tres enfoques descritos en las siguientes secciones. Cada uno de los procedimientos tuvo como entrada los mismos dos conjuntos mencionados y cada uno entregó igualmente dos conjuntos de oraciones, separadas por la orientación de la opinión que en ellas existe.

3.2.7.1. Oraciones con adjetivos o adverbios

De acuerdo con trabajo previo citado en [2] y descrito en [3], los adjetivos y adverbios son indicadores de subjetividad. Por lo tanto, las oraciones que tuvieron al menos uno de estos fueron seleccionadas para entrenar el clasificador.

Se tokenizó cada oración de cada grupo y posteriormente se les aplicó un etiquetado PoS.

Una vez etiquetados los tokens, se filtraron dependiendo de si contenían o no un adjetivo o un adverbio (permanecieron solo las oraciones que sí contenían alguno). Las etiquetas buscadas fueron JJ, JJR, JJS y VBG.

Después del filtrado, se obtuvieron las siguientes cantidades de oraciones:

- Conjunto de oraciones negativas: 6,387.
- Conjunto de oraciones positivas: 8,024.

3.2.7.2. Oraciones con disparadores de presuposición

Una presuposición es el conocimiento implícito de que una declaración debe ser mutuamente conocida o asumida por el hablante y el destinatario, para que la declaración sea considerada apropiada dentro del contexto [45, 46]

Un disparador de presuposición es una construcción que señala la existencia de una presuposición en una declaración.

En [45] se ejemplifican los tipos de disparadores usados para filtrar las oraciones que los contienen de las que no. Esto se hizo de nueva cuenta con la ayuda de expresiones regulares.

Después de realizar este filtrado se obtuvieron las siguientes cantidades:

- Conjunto de oraciones negativas: 1,049.
- Conjunto de oraciones positivas: 2,758.

3.2.7.3. Oraciones con disparadores o adjetivos o adverbios

Como tercer y último enfoque, se tomaron los enunciados que tenían disparadores o adjetivos y adverbios.

Los conjuntos obtenidos fueron:

- Conjunto de oraciones negativas: 7,436.
- Conjunto de oraciones positivas: 10,782.

3.2.8. Validación cruzada con 10 pliegues

Cada línea de experimentación entregó a esta etapa los dos conjuntos de oraciones, uno clasificado como de orientación positiva y otro clasificado como de orientación positiva. Estas clases, positiva y negativa, son las que el clasificador bayesiano ingenuo asignará después de ser entrenado.

En total, se tuvieron cinco pares de conjuntos (diez conjuntos en total). Uno por cada tipo de experimento:

1. Enunciados agrupados con FNM.

2. Enunciados subjetivos elegidos por presencia de adjetivos o adverbios.

3. Enunciados subjetivos elegidos por presencia de disparadores de presuposición.

4. Enunciados subjetivos elegidos por presencia de disparadores de presuposición o adjetivos o adverbios. Este experimento es la unión de los dos anteriores.

5. Todos los enunciados obtenidos en la etapa Separación de reseñas por orientación. Este entrenamiento fue realizado como método de control.

Los subprocesos de entrenamiento, pruebas y evaluación se engloban en el proceso de validación cruzada, ya que esta afecta a los tres procesos.

Como se explicó con anterioridad, en el capítulo dos sección 2.3.2.1.1, durante la validación cruzada con k - pliegues (en este caso $k = 10$) se dividen los datos disponibles (las oraciones) en k pliegues y se entrenan k clasificadores, cada uno con $k - 1$ (en este caso 9) diferentes pliegues. Posteriormente, se prueba cada clasificador con el pliegue con el que no se entrenó.

De esta forma, cada uno de los conjuntos de los cinco pares se dividió en 10 pliegues del mismo tamaño. Se tomaron solo 6,000 enunciados por polaridad y en cada experimento estos se seleccionaron aleatoriamente.

En la figura 3.8 se ilustra lo descrito para un solo conjunto de oraciones.

Recordando el capítulo dos sección 2.3.1.1, el clasificador bayesiano ingenuo requiere de información ya etiquetada con anterioridad para poder predecir la clase de un ejemplo nuevo no visto antes.

El clasificador recibe nueve pliegues en cada entrenamiento, obtiene frecuencias relativas de las palabras de los enunciados de cada uno de estos pliegues y un valor suavizado. Este valor se asigna durante las pruebas a las palabras que no se hayan encontrado en el entrenamiento y que por lo tanto no tienen frecuencia relativa conocida.

Cuando ya se ha entrenado el clasificador, entonces se prueba. La prueba consiste en predecir la clase para los enunciados del pliegue de pruebas (enunciados no vistos antes por el clasificador) y obtener las medidas de desempeño que evalúan el comportamiento del clasificador.

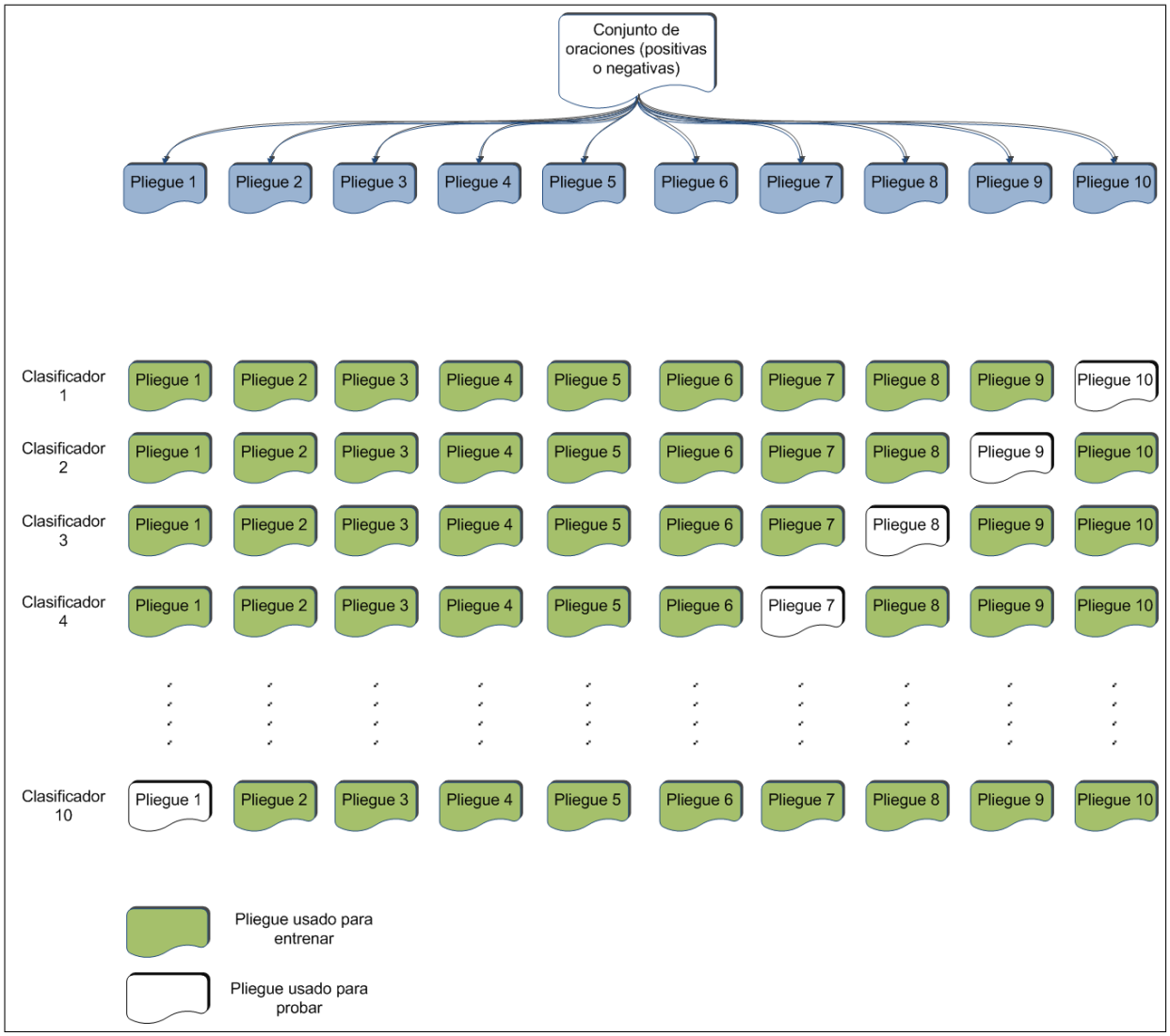

Figura 3.8: Validación cruzada con 10 pliegues - La figura ejemplifica como se realizó el entrenamiento y prueba del clasificador. Se toman solo $k-1$ pliegues para entrenar por cada clasificador y se prueba con los sobrantes. Todos los enunciados son utilizados.

Las medidas de precisión y error y también las medidas de la matriz de confusión sirven para evaluar el desempeño del clasificador y así determinar la efectividad de las técnicas usadas, o saber si cambiando algún parámetro o agregando alguna característica el sistema mejora su desempeño.