

PONTIFICIA UNIVERSIDAD CATOLICA DE VALPARAISO

FACULTAD DE INGENIERÍA

ESCUELA DE INGENIERIA INFORMATICA

DESARROLLO DE UN SISTEMA DE CLASIFICACIÓN BINARIA AUTOMÁTICA DE NOTICIAS CON MÁQUINAS DE APRENDIZAJE

ALBERTO ALEJANDRO HOLTS COREY

CLAUDIO LUIS RIQUELME JEREZ

Profesor Guía: Rodrigo Alfaro Arancibia

Profesor Correferente: Guillermo Cabrera Guerrero

INFORME FINAL DEL PROYECTO

PARA OPTAR AL TÍTULO PROFESIONAL DE

INGENIERO DE EJECUCIÓN EN INFORMÁTICA

OCTUBRE 2010

Agradezco a mis padres por la educación, por el cariño
y todo el apoyo que me han dado,
a mi novia Violeta por el apoyo
y el amor incondicional,
a toda mi familia,
a mi hermano Diego,
y a mi compañero Alberto.

Claudio Riquelme

Agradezco a mi familia y amigos que me
han acompañado a lo largo de esta etapa, en especial

a mi madre Cristina,

a mi hermano Christian,

a mi abuela Raquel

y a mi compañero Claudio

Alberto Holts

INDICE

1 Descripción del proyecto.....	1
1.1 Introducción.....	1
1.2 Definición de objetivos.....	2
1.2.1 Objetivos generales	2
1.2.2 Objetivos específicos	2
1.3 Plan de trabajo	2
2 Clasificación de texto.....	4
2.1 Definición	4
2.2 Clasificación de etiquetado único vs etiquetado múltiple.....	4
2.3 Clasificación centrada en las categorías vs centrada en los documentos.....	5
2.4 Clasificación dura vs clasificación con ranking	5
2.5 Aplicaciones de la clasificación de textos.	5
2.5.1 Indexación automática para los sistemas booleanos de recuperación de información	6
2.5.2 Filtro de textos.....	6
2.5.3 Desambiguación de palabras.....	6
2.5.4 Clasificación jerárquica de páginas web	7
3 Máquinas de aprendizaje.....	8
3.1 Definición	8
3.2 Aplicaciones de máquinas de aprendizaje	8
3.2.1 Aprendizaje por asociaciones.....	8
3.2.2 Clasificación.....	8
3.2.3 Regresión	9
3.3 Tipos de aprendizaje.....	9
3.3.1 Aprendizaje no supervisado	9
3.3.2 Aprendizaje supervisado	10
3.3.3 Aprendizaje semi-supervisado	10
3.3.4 Aprendizaje por refuerzo	10
3.4: Uso de máquinas de aprendizaje para la clasificación de texto	11

3.4.1 Enfoques propuestos	11
3.4.1.1 Word level	14
3.4.1.2 Sub-Word level.....	14
3.4.1.3 Multi-Word level	15
3.4.1.4 Semantic level	15
3.4.2 Representación de texto	16
3.4.3 Modelo de espacio vectorial.....	16
3.4.4 Reducción de la dimensionalidad	19
3.4.4.1 Eliminación de palabras comunes (Stopwords).....	19
3.4.4.2 Reducción a la raíz (Stemming)	19
3.4.4.3 Document Frequency Thresholding.....	20
3.4.4.4 Información Mutua (Mutual Information, Information Gain)	20
3.4.4.5 χ^2 -statistic	20
3.4.4.6 Similitud entre vectores.....	20
3.4.5 Clasificadores de texto.....	21
3.4.5.1 Clasificadores con reglas de decisión	21
3.4.5.2 Árboles de decisión.....	22
3.4.5.3 Métodos en línea.....	22
3.4.5.4 Algoritmo de Rocchio	23
3.4.5.5 Algoritmo del vecino más próximo y variantes.....	23
3.4.5.6 Redes neuronales	24
3.4.5.7 Naive Bayes.....	24
3.4.5.7.1 Introducción al aprendizaje bayesiano	24
3.4.5.7.2 Teorema de Bayes	25
3.4.5.7.3 Naive Bayes	27
3.4.5.8 Máquina de vectores de soporte (SVM).....	28
3.4.5.8.1 Modelo lineal de vectores de soporte	29
3.4.5.8.2 Modelo no lineal de vectores de soporte	31
3.4.6 Evaluación de los clasificadores.....	34

3.4.6.1	Introducción	34
3.4.6.2	Medidas para la efectividad en categorización de texto	34
3.4.6.2.1	Precision y Recall.....	34
3.4.6.2.2	Microaveraging	35
3.4.6.2.3	Macroaveraging	35
3.4.6.2.4	Precision/Recall Breakeven	36
3.4.6.2.5	F _β -Measure.....	36
4	Pruebas sobre algoritmos escogidos	37
4.1	Introducción.....	37
4.2	Colecciones de prueba.....	37
4.2.1	Reuters-21578	37
4.2.2	WebKB	38
4.2.3	Ohsumed	38
4.3	Caso de estudio	38
4.3.1	Introducción	38
4.3.2	Obtención del conjunto de datos para la clasificación.....	38
4.3.3	Representación de texto de los artículos obtenidos	41
4.3.4	Uso de las herramientas de LIBSVM	43
4.4	Pruebas sobre colección Reuters-21578	48
4.5	Conclusiones de las pruebas	49
4.5.1	Comparación Caso de estudio-Reuters 21578	51
5	Definición del sistema.....	53
5.1	Introducción.....	53
5.2	Elección metodología de desarrollo	53
5.3	Especificación de requerimientos.....	54
5.3.1	Requerimientos funcionales	54
5.3.1.1	Requerimientos Usuario.....	54
5.3.1.2	Caso de uso general del sistema	55
5.3.1.3	Caso de uso “Configurar Pre- Entrenamiento”	56
5.3.1.4	Caso de uso “Editar Categorías”	57

5.3.1.5 Caso de uso “Clasificar Documentos”	57
5.3.1.6 Caso de uso “Editar Documentos”	58
5.3.1.7 Caso de uso “Configurar Entrenamiento”	59
5.3.2 Requerimientos no funcionales	59
5.4 Modelado del sistema	60
5.4.1 Diagrama de clases	60
5.4.2 Diagramas de secuencia	61
5.4.2.1 “Configurar Pre- Entrenamiento”	61
5.4.2.2 “Editar Categorías”	61
5.4.2.3 “Clasificar Documentos”	62
5.4.2.4 “Configurar Entrenamiento Bayes”	62
5.4.2.5 “Configurar Entrenamiento SVM”	63
5.5 Descripción del software clasificador	63
5.5.1 Interfaz Pre-Entrenamiento	64
5.5.2 Interfaz Entrenamiento	65
5.5.3 Interfaz Clasificar	70
5.5.4 Interfaz Editar Categorías	72
5.5.5 Interfaz Editar Documentos	73
5.5.6 Diagrama de estado de las interfaces del software	74
6 Caso de estudio utilizando el software construido	75
7 Conclusiones	81
8 Referencias	82

INDICE DE FIGURAS

Figura 1.1 Plan de Trabajo Primer Semestre	2
Figura 1.2 Plan de Trabajo Segundo Semestre	3
Figura 2.1 Jerarquía de páginas web	7
Figura 3.1 Representación típica del BoW	14
Figura 3.2 Porcentaje de similitud entre palabras usando n-gramas	15
Figura 3.3 Términos Relevantes	18
Figura 3.4 Fórmulas para la obtención de la similitud entre vectores.....	21
Figura 3.5 Reglas o cláusulas [15].....	22
Figura 3.6 Árbol de decisión equivalente a las reglas.....	22
Figura 3.7 Representación gráfica del KNN	23
Figura 3.8 Red Neuronal	24
Figura 3.9 Frontera de decisión lineal.....	29
Figura 3.10 Infinitos hiperplanos	29
Figura 3.11 Vectores de Soporte	30
Figura 3.12 Representación visual de errores de entrenamiento	31
Figura 3.13 Representación visual de mapeo	31
Figura 3.14 Mapeo polinomial	32
Figura 3.15 Kernel Polinomial y RBF	32
Figura 3.16 Truco del Kernel	33
Figura 3.17 Comparación de métodos usando kernel lineal[4]	34
Figura 3.18 Tabla de Contingencia para i categorías [15].....	35
Figura 3.19 Tabla de contingencia global para todo el conjunto de categorías [15].....	36
Figura 4.1 Influencia del tamaño en los conjuntos de training y test [8]	39
Figura 4.2 Enmarcado en cuadro rojo el acceso a los distintos canales	39
Figura 4.3 Código fuente de RSS de la tercera en XML.....	40
Figura 4.4 Direcciones de los canales almacenados en la Base de Datos	40
Figura 4.5 Artículos almacenados en la Base de Datos	41
Figura 4.6 Artículos en minúsculas, sin tildes ni signos de puntuación.....	42
Figura 4.7 Artículos sin stopwords y con la representación del BoW en “formato” LIBSVM.....	42

Figura 4.8 Formato del archivo que almacena la representación de cada artículo	43
Figura 4.9 Escalamiento de datos	43
Figura 4.3 Datos de entrenamiento originales	44
Figura 4.4 Datos de entrenamiento escalados	44
Figura 4.5. Comparación de clasificadores con y sin overfitting.....	45
Figura 4.6 Cross-validation	46
Figura 4.7 Opciones de LIBSVM	46
Figura 4.8 Proceso completo de clasificación	47
Figura 4.9 Resultados Accuracy SVM Set La Tercera	47
Figura 4.10 Resultados Accuracy Bayes La Tercera	48
Figura 4.11 Resultados Accuracy Svm Set Reuters.....	49
Figura 4.12 Resultados Accuracy Svm	50
Figura 5.1 Caso de uso General del Sistema	55
Figura 5.2 Caso de uso “Configurar Pre- Entrenamiento”	56
Figura 5.3 Caso de uso “Editar Categorías”	57
Figura 5.4 Caso de uso “Clasificar Documentos”	57
Figura 5.5 Caso de uso “Editar Documentos”	58
Figura 5.6 Caso de uso “Configurar Entrenamiento”	59
Figura 5.7 Diagrama de clases.....	60
Figura 5.8 “Configurar Pre- Entrenamiento”	61
Figura 5.9 “Editar Categorías”	61
Figura 5.10 “Clasificar Documentos”	62
Figura 5.11 “Configurar Entrenamiento Bayes”	62
Figura 5.12 “Configurar Entrenamiento SVM”	63
Figura 5.13 Interfaz Pre-Entrenamiento.....	64
Figura 5.14 Interfaz Entrenamiento	65
Figura 5.15 Parámetros sugeridos por la cross validation	66
Figura 5.16 Interfaz Cargar Representación.....	67
Figura 5.17 Entrenamiento Finalizado	68
Figura 5.18 Resultados de la evaluación	69
Figura 5.19 Interfaz Clasificar	70

Figura 5.20 Pantalla Clasificación final	71
Figura 5.21 Interfaz Editar Categorías	72
Figura 5.22 Interfaz Editar Documentos	73
Figura 5.23 Diagrama de estado de las interfaces del software	74
Figura 6.1 Resultados F_1 para Reuters usando SVM	76
Figura 6.2 Resultados F_1 para Reuters usando Naive Bayes	76
Figura 6.3 Resultados F_1 para La Tercera usando SVM	77
Figura 6.4 Resultados F_1 para La Tercera usando Naive Bayes	78
Figura 6.5 Diagrama de Boxplot F_1 para Reuters usando SVM	78
Figura 6.6 Diagrama de Boxplot F_1 para Reuters usando Bayes	79
Figura 6.7 Diagrama de Boxplot F_1 para La Tercera usando SVM	79
Figura 6.8 Diagrama de Boxplot F_1 para La Tercera usando Bayes	80

INDICE DE TABLAS

Tabla 4.1 Tabla Resultados Accuracy SVM Set La Tercera.....	47
Tabla 4.2 Resultados Accuracy Bayes Set La Tercera.....	48
Tabla 4.3 Resultados Svm Set Reuters	49
Tabla 4.4 Resultados Accuracy Svm	49
Tabla 4.5 Evaluación SVM	51
Tabla 4.6 Evaluación Bayes	51
Tabla 6.1 Resultados F_1 para Reuters usando SVM	75
Tabla 6.2 Resultados F_1 para Reuters usando Naive Bayes	76
Tabla 6.3 Resultados F_1 para La Tercera usando SVM	77
Tabla 6.4 Resultados F_1 para La Tercera usando Naive Bayes	77

Resumen

El almacenamiento de documentos en formato digital y el enorme y creciente volumen de información existente en la web traen consigo la necesidad de recopilarla de tal manera que su acceso sea organizado y rápido, la mejor manera de alcanzarlo es realizando una clasificación sobre la información recopilada.

La tarea de realizar una clasificación de forma manual (por parte de expertos humanos) es un proceso tedioso, lento y costoso, es por ello que surge la problemática de desarrollar una clasificación automática de texto.

El objetivo de esta investigación es resolver el problema de la clasificación automática aplicada al caso específico de la clasificación de noticias, haciendo una clasificación binaria mediante el uso de máquinas de aprendizaje.

El presente informe de trabajo de título detalla el proceso de investigación llevado a cabo para desarrollar una herramienta que permita entrenar una máquina de aprendizaje y luego ser usada para clasificar documentos. Para ello se describen las definiciones de máquinas de aprendizaje y clasificación de texto. Se exponen las distintas técnicas para clasificar y se presentan los resultados de pruebas realizadas para optar por la mejor técnica. Finalmente se describe el sistema que realiza esta clasificación y se presentan resultados de su clasificación.

Palabras Clave: Clasificación de texto, Máquinas de aprendizaje, SVM, Bayes ingenuo.

Abstract

The increased availability of documents in digital format available on the web contain useful information for different purposes, this carries an important need to organize them.

The classification task achieved manually (by humans experts), is tedious, slow and expensive, that's why the problem is the developing of an automatic text classification.

The goal of this research is to solve the problem of automatic classification applied to the specific case of classifying news by making a binary classification using machine learning techniques.

This report details the investigation process carried out to develop a tool to train a machine learning algorithm and then be used to classify documents. Describes the definition of machine learning and text classification. Describes the various techniques for automated text classification and shows the results of testing, to choose the best techniques. Finally describes the system that performs this classification and the results of it.

Keywords: Text Classification, Machine Learning, SVM, Naïve Bayes.

1 Descripción del proyecto

1.1 Introducción

La Recuperación de Información, es un tema preponderante en el campo de los sistemas de información, ya que la cantidad de documentos crece descontroladamente, haciendo necesario clasificarlos con el fin de ayudar una posible búsqueda de contenido y tener un acceso fácil y organizado a ellos.

La Categorización de Texto, es unas de las tareas de la Recuperación de Información, consistente básicamente en etiquetar textos en lenguaje natural, en categorías temáticas predefinidas de una colección ya existente. [16].

Si bien una clasificación de textos se puede realizar manualmente, es deseable alcanzar un nivel de automatización de estos procedimientos. Esta tarea es conocida como clasificación automática de documentos y es usada en diversos contextos (la propia clasificación de documentos temáticos, tales como documentos científicos o artículos de prensa, como también otro tipo de aplicaciones tales como la herramienta de filtrado de correo basura).

El hecho que la tecnología actual permite tener estos documentos en un formato digital accesible para máquinas, hace posible que la tarea de clasificarlos pueda ser automatizada, al ser una tarea muy necesaria, diversas investigaciones se han realizado, por lo que existen distintos métodos para llevarlo a cabo, el más popular es el uso de Máquinas de Aprendizaje.

Las Máquinas de Aprendizaje corresponden a un área de la Inteligencia Artificial que se ocupa del desarrollo de técnicas que permitan aprender a partir de la experiencia, y a través de un aprendizaje inductivo extraer reglas a partir de un conjunto de elementos de ejemplo.

Por lo tanto, la intención de este proyecto es investigar los métodos de máquinas de aprendizaje más utilizados en la categorización de texto, basados en la literatura, con el fin futuro de aplicarlo, y realizar comparaciones de los métodos escogidos.

Se estima conveniente abordar los siguientes tópicos como base teórica en la realización del proyecto: el estado del arte de la categorización de texto, y los conceptos de categorización de texto y máquinas de aprendizaje.

El informe consta de seis capítulos más conclusiones, en el primer capítulo se expone la problemática de la clasificación automática, los objetivos y planificación del trabajo de investigación propuesto, en el segundo y tercer capítulo se define la clasificación de texto, máquinas de aprendizaje y su uso para resolver el problema de la clasificación automática, en el cuarto capítulo se presenta un caso de estudio con el fin de realizar pruebas sobre las técnicas estudiadas, luego en el quinto capítulo se describe el sistema que se desarrollará para efectuar la clasificación de noticias, a continuación en el sexto capítulo se realizan nuevas pruebas sobre datos de benchmark y caso de estudio, haciendo uso del sistema construido .

Finalmente con los resultados arrojados por el estudio se presentan las conclusiones finales del trabajo de investigación realizado.

1.2 Definición de objetivos

1.2.1 Objetivos generales

- Desarrollar un sistema clasificador binario automático de noticias, usando técnicas de Máquinas de Aprendizaje.

1.2.2 Objetivos específicos

- Investigar distintas técnicas de representación de texto, algoritmos de aprendizaje, clasificadores y evaluación de clasificadores.
- Comparar el resultado obtenido de las distintas técnicas escogidas, con el fin de seleccionar la que mejor se adecúa al problema.
- Desarrollar un sistema que permita clasificar noticias automáticamente y despliegue los resultados obtenidos para el usuario.

1.3 Plan de trabajo

	Nombre de tarea	Duración	Comienzo	Fin
1	Proyecto I	78 días?	mié 8/5/09	vie 11/20/09
2	Investigación Inicial	28 días	mié 8/5/09	vie 9/11/09
3	Investigación Estado del Arte	8 días	mié 8/5/09	vie 8/14/09
4	Lectura Papers propuestos	10 días	lun 8/17/09	vie 8/28/09
5	Investigación General TC y ML	10 días	lun 8/31/09	vie 9/11/09
6	Preparación Informe de Avance	10 días?	lun 9/14/09	vie 9/25/09
7	Redacción Inicial	5 días	lun 9/14/09	vie 9/18/09
8	Revisión Informe	1 día?	lun 9/21/09	lun 9/21/09
9	Corrección de Errores	3 días	mar 9/22/09	jue 9/24/09
10	Entrega Informe de Avance	1 día?	vie 9/25/09	vie 9/25/09
11	Investigación y Pruebas	25 días	lun 9/28/09	vie 10/30/09
12	Investigación RT	7 días	lun 9/28/09	mar 10/6/09
13	Investigación Bayes y SVM	7 días	lun 9/28/09	mar 10/6/09
14	Adaptación Algoritmos	18 días	mié 10/7/09	vie 10/30/09
15	Pruebas sobre Algoritmos	13 días	mié 10/14/09	vie 10/30/09
16	Informe Final y Definición Sistema	15 días?	lun 11/2/09	vie 11/20/09
17	Redacción Inicial	9 días	lun 11/2/09	jue 11/12/09
18	Definición Alcance del Sistema	9 días	lun 11/2/09	jue 11/12/09
19	Evaluación de Técnicas	9 días	lun 11/2/09	jue 11/12/09
20	Revisión Informe	1 día?	vie 11/13/09	vie 11/13/09
21	Corrección de Errores	4 días	lun 11/16/09	jue 11/19/09
22	Entrega Informe Final	1 día?	vie 11/20/09	vie 11/20/09

Figura 1.1 Plan de Trabajo Primer Semestre















	 Nombre de tarea	Duración	Comienzo	Fin
1	 Proyecto II	75 días?	lun 3/15/10	vie 6/25/10
2	 Prueba sobre Benchmarks y Preparación Informe de Avance	30 días	lun 3/15/10	vie 4/23/10
3	 Proceso de Reuters y Representación	10 días	lun 3/15/10	vie 3/26/10
4	Redacción Inicial Informe de Avance	10 días	lun 3/29/10	vie 4/9/10
5	 Evaluación Clasificadores	5 días	lun 4/12/10	vie 4/16/10
6	Revisión y Corrección Informe	4 días	lun 4/19/10	jue 4/22/10
7	 Entrega Informe de Avance	1 día	vie 4/23/10	vie 4/23/10
8	 Diseño y Construcción de Software	30 días	lun 4/26/10	vie 6/4/10
9	 Diseño	5 días	lun 4/26/10	vie 4/30/10
10	Investigación Bayes y SVM	20 días	lun 5/3/10	vie 5/28/10
11	Adaptación Algoritmos	5 días	lun 5/31/10	vie 6/4/10
12	 Pruebas de Software y Preparación Informe Final	15 días?	lun 6/7/10	vie 6/25/10
13	 Redacción Inicial	5 días	lun 6/7/10	vie 6/11/10
14	 Pruebas de Software	4 días	lun 6/14/10	jue 6/17/10
15	 Revisión SW	0 días	vie 6/18/10	vie 6/18/10
16	 Corrección de Errores	4 días	lun 6/21/10	jue 6/24/10
17	 Entrega Informe Final	1 día?	vie 6/25/10	vie 6/25/10

Figura 1.2 Plan de Trabajo Segundo Semestre

El plan de trabajo expuesto en las Figuras 1.1 y 1.2 corresponde a las etapas y tareas definidas correspondientes a los periodos establecidos para Proyecto I, comenzó el 5 de Agosto de 2009 hasta 20 de Noviembre del 2009, y Proyecto II, período del 15 de Marzo del 2010 a 25 de Junio de 2010.

Las tareas asignadas a Proyecto I, en primera instancia, estuvieron avocadas a la investigación de la clasificación de texto automática y su resolución haciendo uso de máquinas de aprendizaje, luego la culminación consistía en probar algoritmos basados en alguna de las técnicas escogidas de la etapa anterior y una definición del alcance del sistema a construir.

Para Proyecto II se definió el sistema a construir y se elaboraron pruebas con benchmark conocidos con el objeto de ser realizadas con el software propuesto, por lo que una vez implementado el sistema el paso posterior fue efectuar pruebas haciendo uso del software desarrollado.

2 Clasificación de texto

2.1 Definición

La clasificación (o categorización) de texto se define como la tarea de asignar un valor booleano para cada par (d_j, c_i) perteneciente a $D \times C$, donde D es el dominio de documentos y $C = \{c_1, \dots, c_j\}$ es una serie de categorías predefinidas [15]. Un valor de verdad (T) asignado a (d_j, c_i) indica que el texto d_j clasifica dentro de la categoría c_i , mientras que un valor de falsedad (F) indica que el documento no clasifica dentro de la categoría. Más formalmente la tarea de clasificar texto consiste en encontrar una función $\gamma: D \times C \rightarrow \{T, F\}$ llamada clasificadora (regla, hipótesis o modelo) que “coincida lo más posible” (alta efectividad) con la función desconocida $\gamma^*: D \times C \rightarrow \{T, F\}$ que describe cómo los documentos deberían ser clasificados.

Para efectos de la construcción de un algoritmo de clasificación automatizada de texto se asume que las categorías son solo etiquetas simbólicas y no agregan conocimiento alguno que ayude a la construcción del clasificador, además se asume que no se cuenta con ningún conocimiento exógeno, y que la clasificación de los documentos solo se hará con conocimiento endógeno que puede ser extraído del documento en sí, es decir que no se cuenta con conocimiento de metadato como la fecha de publicación, tipo de documento, fuente de publicación, autor etc.

Centrarse solo en conocimiento endógeno implica tratar de clasificar un documento basándose solo en su semántica, lo que resulta subjetivo, por lo que la clasificación de un documento en cierta categoría no puede ser decidida de forma determinista.

2.2 Clasificación de etiquetado único vs etiquetado múltiple.

Se pueden aplicar diferentes restricciones en las categorías dependiendo de los requerimientos de la aplicación, el etiquetado único (o categorías sin superposición) se da cuando cada documento es asignado a solo una categoría, en cambio cuando un documento se asigna a más de una categoría se conoce como clasificación de texto con etiquetado múltiple (o categorías con superposición) lo que permite conciliar de alguna manera la clasificación hecha por humanos expertos que difieren al clasificar un texto que puede caer en más de una categoría. Un caso especial de etiquetado único es el de la clasificación binaria en donde un documento d debe ser asignado a una categoría c o a su complemento \bar{c} .

Teóricamente el caso de clasificación binaria es más general que el de etiquetado múltiple, por lo que un algoritmo para clasificación binaria podría ser usado para la clasificación con etiquetado múltiple previa transformación del problema de n categorías en n problemas independientes de clasificación binaria. Esto requiere, sin embargo, que las categorías sean totalmente independientes y excluyentes entre sí. El caso contrario no es factible, un algoritmo para clasificación de etiquetado múltiple no puede ser usado para clasificación de etiquetado único ni para el binario.

El caso de clasificación binaria parece ser el más adecuado para la investigación, ya que las principales aplicaciones de categorización de texto consisten en problemas binarios, por ejemplo filtros correos, clasificándolos como “spam” o “no spam”. Y como ya se explicó

resolviendo el caso de clasificación binaria implica resolver también el de etiquetado múltiple. La clasificación binaria además es más simple de ilustrar y solo un caso especial de la de etiquetado único.

2.3 Clasificación centrada en las categorías vs centrada en los documentos.

Una vez construido el clasificador de texto existen dos maneras de usarlo. Dado un documento a clasificar, se buscan todas las categorías a las cuales se podría asignar, esta forma de afrontar la clasificación es centrada en la categoría (document-pivoted categorization, DPC); por otro lado, dada una categoría se podría buscar todos los documentos que calzan en ella esta forma es centrada en las categorías (category-pivoted categorization, CPC). Es importante esta obvia distinción tomando en cuenta que el total de las categorías y los documentos no estarán disponibles desde el principio, y es relevante la elección en el método usado para la construcción del clasificador, ya que algunos clasificadores cuentan con una inclinación definida hacia uno u otro estilo de clasificación.

Clasificación centrada en los documentos es apropiada cuando los documentos a clasificar no están disponibles desde el principio sino que están disponibles uno a la vez en un periodo de tiempo (por ejemplo al clasificar emails). La clasificación centrada en las categorías en cambio es apropiada cuando una nueva categoría es agregada luego de haber clasificado ya una cierta cantidad de documentos o cuando estos documentos necesitan reclasificarse bajo las categorías existentes más las nuevas.

Aunque existen técnicas de clasificación que se aplican solo a un estilo (CPC o DPC) la mayoría de las técnicas que luego se discutirán permiten la construcción de clasificadores capaz de trabajar con ambos.

2.4 Clasificación dura vs clasificación con ranking.

Una clasificación de texto normal requiere una decisión de verdadero (T) o falso (F) para cada par (d, c) , pero esta clasificación puede tener distintos requerimientos. Por ejemplo dado un documento d , el sistema podría simplemente asignar un número para “rankear” el grado de pertenencia a las categorías $C = \{c_1, \dots, c_j\}$ sin tomar una decisión “dura”. Esta lista rankeada podría ser de gran utilidad para humanos expertos para tomar la decisión final acerca de la clasificación, considerando solo los documentos que más se acerquen a las categorías dadas. Así mismo dada una categoría c_i el sistema podría rankear los documentos de D acorde a la pertenencia a c_i . Estas dos modalidades son llamadas ranking de documentos y de categoría respectivamente y son claramente una contraparte a CPC y DPC.

2.5 Aplicaciones de la clasificación de textos.

La clasificación automática de texto data de principio de los 60, con el trabajo de clasificación de texto probabilística de Maron (1961). Desde entonces ha sido usada en distintas aplicaciones, algunas de ellas hacen uso de otras ciencias como por ejemplo combinar el reconocimiento de voz con la clasificación de texto.

A continuación se nombrarán algunas de estas aplicaciones, las más importantes.

2.5.1 Indexación automática para los sistemas booleanos de recuperación de información

Esta fue uno de los primeros sistemas donde se usaron los clasificadores de texto, la indexación de documentos para IR (Information Retrieval) depende de un diccionario controlado, en donde a cada documento se le asigna una palabra o frase clave que describe su contenido, estas palabras se sacan del diccionario que contiene temáticas jerarquizadas en un tesoro. Usualmente esta indexación era hecha por expertos entrenados para ello y era muy costosa de realizar.

Si cada palabra del diccionario controlado es vista como una categoría, esta indexación es un claro caso de clasificación de texto y puede ser afrontada con técnicas automáticas, siendo la categorización basada en documentos la mejor opción para ello.

La indexación de documentos con un diccionario se relaciona fuertemente con la generación automática de metadato. Este metadato se usa en bibliotecas para etiquetar documentos según distintos aspectos y así poder organizarlos, lo que se puede considerar otra aplicación de clasificación de texto. Esta generación puede ser vista como un problema de indexación y por consiguiente un problema de clasificación de texto.

2.5.2 Filtro de textos

El filtro de texto es la actividad de clasificar una colección de textos que van llegando dinámicamente, por ejemplo los textos que llegan a un periódico para ser publicado, estos textos llegan por distintas vías de comunicación y tienen que ser clasificados para saber su importancia y prioridad de publicación. En este caso el sistema de filtro debe poder bloquear la información que no debería ser publicada según algún criterio. La filtración de texto puede ser vista como un caso de clasificación de texto de etiquetado único, por ejemplo clasificar documentos en dos categorías excluyentes como la información relevante e irrelevante para el periódico. Además un sistema de filtros puede realizar más clasificaciones internas dentro de las primeras categorías, por ejemplo clasificar los documentos relevantes según las temáticas de las noticias.

La filtración de texto tiene tradición y data desde los 60's cuando existían diversos sistemas de información con variados grados de automatización, donde se daban casos donde se debía filtrar mucha información. La explosión de disponibilidad de información en formato digital, particularmente en la internet, ha hecho importante estos sistemas, y son usados en diversos contextos como la creación de periódicos en línea personalizados, bloqueo de email basura, etc.

2.5.3 Desambiguación de palabras

La desambiguación de palabras (Word sense disambiguation, WSD) se refiere a la actividad de encontrar el sentido de una palabra en particular, cuando su significado es ambiguo en un texto. Por ejemplo la palabra banca, tiene al menos tres significados, una entidad financiera, mueble largo de estructura sencilla en el que pueden sentarse varias personas a la vez, o un conjunto de peces. WSD es ocupado en un importante número de aplicaciones, incluyendo el entendimiento de lenguaje natural.

WSD es un caso de clasificación de etiquetado único, y es solo un ejemplo del problema de entender el lenguaje natural y su ambigüedad, otras instancias de este problema son la corrección de ortografía según el contexto, etiquetado de discursos, y elección de palabras en máquinas de traducción entre otros.

2.5.4 Clasificación jerárquica de páginas web

La clasificación automática de documentos toma gran importancia en sus usos posibles en aplicaciones de internet. Una de estas es la clasificación de páginas web en distintas categorías, que puedan ser jerarquizadas y agregadas en catálogos comerciales y en portales populares de internet. De esta manera los motores de búsqueda encontrarán más fácil el sitio y será más fácil navegar dentro de él para encontrar, por ejemplo, un categoría particular de interés.

La clasificación de páginas web es obviamente un ventaja, ya que la clasificación manual de grandes cantidades de contenidos es infactible. A diferencia de las otras aplicaciones, en este caso se quiere que cada categoría sea habitada por una cantidad x de documentos, por lo que escoger clasificación basada en categoría sería lo adecuado, tomando en cuenta que se desea permitir agregar nuevas categorías y eliminar las obsoletas cuando se desee.

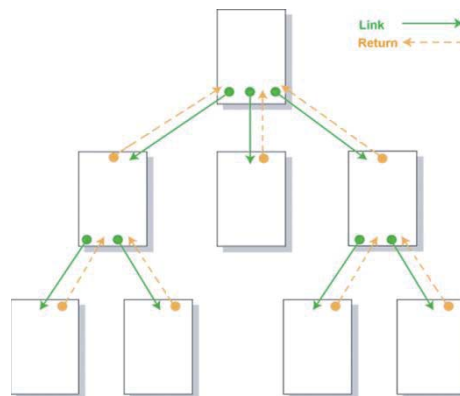


Figura 2.1 Jerarquía de páginas web

3 Máquinas de aprendizaje

3.1 Definición

Máquina de aprendizaje (Machine Learning) o aprendizaje automático es una disciplina científica, rama de la inteligencia artificial, cuyo objetivo es desarrollar técnicas que permitan a las computadoras “aprender”, a través de un proceso inductivo generalizando comportamientos a partir de una base de conocimiento suministrada en forma de ejemplos.

“Machine Learning es programar computadoras para optimizar el desempeño de su criterio usando datos de ejemplo o experiencia pasada” [1]. Se tiene un modelo definido con ciertos parámetros, el aprendizaje es la ejecución de un programa computacional para optimizar los parámetros del modelo usando los datos ejemplo y la experiencia pasada a modo de entrenamiento. El modelo puede ser predictivo para hacer futuras predicciones, descriptivo para ganar conocimiento de los datos o ambos.

Las máquinas de aprendizaje usan teorías de estadística para construir modelos matemáticos para cumplir con la tarea principal de hacer inferencias con algunos datos de muestras, para ofrecer una visión más detallada se expondrán algunas aplicaciones en las que se usa máquinas de aprendizaje.

3.2 Aplicaciones de máquinas de aprendizaje

3.2.1 Aprendizaje por asociaciones

Una aplicación de máquinas de aprendizaje relacionado con la minería de datos, es el análisis de la relación entre distintos datos. Por ejemplo en canastas de compra, en donde se trata de encontrar asociaciones entre productos comprados por los consumidores. Si se aprecia que la gente que compra X típicamente compra también Y , un consumidor que compra X pero no Y es identificado como un consumidor potencial de Y .

La idea es encontrar una regla de asociación, el interés reside en aprender sobre la probabilidad $P(Y/X)$ donde Y es el producto que está condicionado por X . También hacer distinción entre los consumidores para estimar $P(Y/X, D)$ donde D es una serie de atributos de consumidor, por ejemplo, genero, edad, estado civil, etc. Asumiendo que tenemos acceso a esta información de esta manera estimar valores más aproximados a la realidad en función de los nuevos atributos. Esta aplicación puede pensarse en otros contextos, en el caso de una página web, los productos serían links y se podría estimar los links que un usuario podría seguir y usar esta información para pre cargar estas páginas por adelantado para un acceso más rápido.

3.2.2 Clasificación

En problemas donde existen clases de separación, como por ejemplo la distinción de un cliente de alto riesgo con el de uno de bajo riesgo que deben hacer los bancos antes de efectuar un préstamo, se usa la información del problema como datos de entrenamiento para que un clasificador pueda asignar los datos de entrada, en este caso los datos de los nuevos consumidores, a alguna de las dos clases posibles. En algunos casos, en vez de tomar una

decisión del tipo 1 o 0 (alto riesgo, bajo riesgo), se podría calcular la probabilidad de la pertenencia de un cliente a un grupo u otro.

El reconocimiento de patrones es otra aplicación de máquinas de aprendizaje que cae dentro de la clasificación, por ejemplo el reconocimiento de caracteres a través de su representación como imagen, acá se clasifica en múltiples clases, tantas como caracteres se quieran reconocer (el abecedario por ejemplo).

Otro caso es la detección de excepciones, en donde se trata de encontrar instancias que no obedezcan a las reglas dadas, o que no correspondan a una categoría, así se pueden encontrar casos que pueden implicar anomalías, a lo que se les debe poner atención para, por ejemplo, detectar un fraude.

3.2.3 Regresión

Los problemas de regresión son aquellos en los que se espera como salida un valor numérico, no un valor booleano como en la clasificación. Por ejemplo para calcular el valor comercial de un automóvil se tendrá como entrada atributos del automóvil que puedan ser importantes en esta situación como la marca, año, capacidad del motor, kilometraje, y se debe llegar al valor con una función que considere estas características. Otro ejemplo sería un sistema de navegación automatizado de un vehículo, en el que la salida sería el ángulo en que se tienen que girar las ruedas en cada instante para avanzar sin colisionar con obstáculos o desviarse de la ruta. Las entradas en este caso serían proporcionadas por sensores en el auto, cámaras, sistema GPS, etc. Los datos de entrenamiento serían recolectados monitoreando las acciones de un conductor humano.

No se trata de aprender la pertenencia a una clase sino encontrar una función continua que arrojen los valores numéricos correctos. En general los casos de regresión tratan de optimizar una función.

3.3 Tipos de aprendizaje

3.3.1 Aprendizaje no supervisado

En el aprendizaje no supervisado solo existen datos de entrada y el objetivo es encontrar las regularidades en estos datos. Hay estructuras con ciertos patrones en los datos de entrada, patrones que se producen con más frecuencia que otros y se quiere encontrar lo que ocurre normalmente y lo que no, cómo están organizados los datos. No hay categorías previas ni esquemas o cuadros de clasificación establecidos a priori. Los documentos se agrupan en función de ellos mismos, de su contenido, se puede decir que se auto organizan. Es lo que se conoce como clasificación no supervisada o clustering. Y es no supervisada porque se efectúa de forma totalmente automática, sin supervisión o asistencia manual.

Una aplicación interesante de clustering es la compresión de imágenes. En este caso los datos de entrada son píxeles representados como valores RGB (Red Green Blue). El programa de clustering agrupa los píxeles de color similar. Si en una imagen solo hay formas hechas con píxeles con una baja cantidad de colores y se agrupan estos píxeles, el peso de la imagen se reducirá. Si se toma, por ejemplo, una imagen donde cada pixel es de 24 bits para

representar 16 millones de colores (2^{24} colores), pero en realidad la imagen hace uso de 64 colores diferentes, se necesitaría solo 6 bits por pixel ($2^6 = 64$) en vez de 24.

3.3.2 Aprendizaje supervisado

El aprendizaje supervisado produce una función que establece una correspondencia entre las entradas y las salidas deseadas del sistema aprendiendo de los datos de entrenamiento. Estos datos de entrenamiento consisten en pares de, objetos de entrada (típicamente vectores) y los datos de salida deseados. Las salidas de la función pueden ser valores continuos (regresión), o predecir el etiquetado de los datos de entrada (clasificación). La tarea es encontrar el valor de la función para cualquier entrada valida luego de entrenarse con los datos de ejemplo.

El aprendizaje supervisado facilita la recuperación de información, limitando las búsquedas a las clases o categorías que el usuario elige, basándose en el conocimiento intelectual que previamente tiene del tema.

Como ejemplo de aprendizaje supervisado se intentará aprender una clase, C , “Auto deportivo”. Se tiene un conjunto de ejemplos de autos y un grupo de usuarios a los que se encuestará acerca de estos y si los consideran autos deportivos, de esta manera los automóviles identificados como autos deportivos se toman como datos positivos de ejemplo y los otros como ejemplos negativos. Para identificar la clase se encuentra una descripción que compartan todos los ejemplos positivos y ningunos de los negativos. A modo de simplificar el ejemplo se considera que las características que separan los autos deportivos del resto son el poder del motor y el precio, y estas características serán los datos de entrada para entrenar el clasificador, para poder llegar a una función $H(x)$ lo más cercana posible $C(x)$ que clasifique como auto deportivo o no a las nuevas entradas no etiquetadas.

3.3.3 Aprendizaje semi-supervisado

El aprendizaje semi-supervisado es un tipo de algoritmo de machine learning que hace uso de datos, tanto etiquetados como no etiquetados previamente como datos de entrenamientos, típicamente un pequeño número de datos etiquetados y una gran cantidad de datos no etiquetados. Este aprendizaje cae entre el aprendizaje no supervisado y el supervisado siendo una mezcla de los dos.

La obtención de datos previamente etiquetados presenta muchas veces un problema, ya que requiere un grupo de humanos calificados para la etiquetación manual de los ejemplos, muchos investigadores han descubierto que los datos no etiquetados junto con una pequeña cantidad de datos etiquetados aumenta considerablemente la efectividad del aprendizaje por lo que es una excelente opción cuando no se cuenta con muchos recursos.

3.3.4 Aprendizaje por refuerzo

En el aprendizaje por refuerzo los datos de salida son una secuencia de acciones, en donde una acción por sí sola no es importante, lo que es importante la secuencia correcta de acciones para alcanzar una meta y la política que llevó a esa serie de acciones. De esta manera el sistema de Machine Learning debe ser capaz de evaluar estas políticas a partir de buenas secuencias de acciones pasadas, luego de varias rondas de prueba y error, y ser capaz de elegir las mejores políticas que puedan maximizar los beneficios.

Un ejemplo de aprendizaje por refuerzo es el juego de ajedrez en donde un simple movimiento no es importante sino la secuencia de movimientos correctos. Un movimiento es una parte de una buena política de juego. Los juegos son una importante fuente de investigación en inteligencia artificial, ya que son fáciles de describir pero difíciles de jugar al mismo tiempo. Un juego de ajedrez tiene una cantidad pequeña de reglas pero es complejo por la gran cantidad de movimientos posibles en cada instante del juego. El sistema tendrá información de su estado solo al final cuando gane o pierda el juego. Una vez que se tienen buenos algoritmos capaces de aprender a jugar, no sería tan difícil utilizarlos en aplicaciones de más utilidad económica.

Otro ejemplo sería el de un robot situado en un laberinto. El robot puede moverse en una de las cuatro direcciones del compás, ejecutando una secuencia de movimientos para alcanzar la salida. Mientras el robot está en el laberinto no tendrá información de su estado de avance, y solo al llegar a la meta tras una serie de movimientos recibirá una recompensa. En este caso no hay oponentes pero se elegirá la trayectoria que menos demore por lo que se podría decir que el tiempo es contra lo que se juega.

3.4: Uso de máquinas de aprendizaje para la clasificación de texto

3.4.1 Enfoques propuestos

La construcción de un sistema para la clasificación de texto es factible realizarla de forma manual, mediante la creación de conjuntos de reglas, lo que tendría un gran costo humano asociado, siendo además una tarea tediosa, por lo que el enfoque más utilizado en la actualidad consiste en el uso de técnicas de Recuperación de Información y Aprendizaje Automático (AA) para inducir un modelo de clasificación denominado **clasificador**.

Existen varias formas de implementar el aprendizaje automático, se destacarán cuatro propuestas hechas por investigadores que han hecho trabajos con documentos en español (Gómez y Venegas) y referentes a nivel mundial en el tema (Joachims y Sebastiani), que conducen a una idea similar.

La primera propuesta por [5] consiste a grosso modo en lo siguiente, el clasificador se puede obtener siguiendo estos dos pasos:

- Representar el conjunto de documentos (en el caso de este proyecto de título se trabajará con artículos de noticias) manualmente clasificados (la colección de entrenamiento) como vectores de pesos de términos (usando el Modelo del Espacio Vectorial de Salton).
- Entrenar una función de clasificación, que puede ser un conjunto de reglas, un árbol de decisión, etc., es decir se hace uso de una máquina de aprendizaje.

Este enfoque es bastante efectivo para los sistemas de categorización de texto orientados a categorías temáticas (una de las características del proyecto), produciendo clasificadores precisos si se dispone de suficientes datos para el entrenamiento.

Una de las decisiones relevantes en la aplicación de este modelo basado en aprendizaje se debe tomar en la etapa inicial y consiste en definir las unidades de representación o términos, el siguiente paso es producto de la cantidad de elementos que se generen en la etapa anterior, y consiste en realizar una reducción de elementos, para finalmente concluir aplicando la técnica de AA escogida.

La segunda propuesta [16] aborda los siguientes tópicos: La clasificación automática de documentos puede concebirse como un proceso de "aprendizaje matemático-estadístico", durante el cual un algoritmo implementado computacionalmente capta las características que distinguen cada categoría o clase de documentos de las demás, es decir, aquellas que deben poseer los documentos para pertenecer a esa categoría. Estas características no tienen porqué indicar de forma absoluta e inequívoca la pertenencia a una clase o categoría, sino que más bien lo hacen en función de una escala o graduación. De esta forma, por ejemplo, documentos que posean una cierta característica tendrán un factor de posibilidades de pertenecer a determinada clase, de modo que la acumulación de dichas características arrojará un resultado que consiste en un coeficiente asociado a cada una de las clases ya conocidas.

Este coeficiente lo que expresa en realidad es el grado de confianza o certeza de que el documento en cuestión pertenezca a la clase asociada al coeficiente resultante.

Las técnicas empleadas para la clasificación de documentos, son originarias de los métodos clásicos de recuperación de información, siendo el modelo vectorial, la base conceptual para las técnicas de clasificación actuales. El modelo vectorial fue definido inicialmente por Salton (1968) y es ampliamente usado en operaciones de recuperación de información, así como también en operaciones de categorización automática, filtrado de información, etc.

Las máquinas de aprendizaje están muy lejos de ser similar al proceso de aprendizaje o de toma de decisión que llevan a cabo los seres humanos, aun cuando algunos resultados de estos procesos automatizados puedan ser útiles e incluso, en ocasiones, sorprendentes.

La tercera proposición investigada es la publicada por [15], trata lo siguiente: En el enfoque de clasificación de texto usando máquinas de aprendizaje, un proceso inductivo general (también llamado learner) automáticamente construye un clasificador para una categoría c_i observando las características de una serie de documentos que han sido previamente clasificadas manualmente bajo c_i o $\neg c_i$ por un experto de dominio; de esas características, el proceso inductivo aprende las características que un documento nuevo debe tener en orden a ser clasificado bajo la categoría c_i . Usando la terminología propia de las máquinas de aprendizaje, el problema de clasificación es una actividad de aprendizaje supervisado, ya que el proceso de aprendizaje es conducido, o "supervisado", por el conocimiento de las categorías y las etapas de entrenamiento que pertenecen a él.

Y en el ámbito de la implementación propone un conjunto de entrenamiento, prueba y validación (Training, Test, Validation), en primer lugar este enfoque depende de la existencia de un corpus inicial $\Omega = \{d_1, \dots, d_{|\Omega|}\}$ de documentos previamente clasificados bajo el mismo conjunto de categorías $C = \{c_1, \dots, c_{|C|}\}$ con cual el sistema deberá operar. Esto significa que los valores de la función final $\Phi: D \times C \rightarrow \{T, F\}$ son conocidos para cada par $\langle d_j, c_i \rangle \in \Omega \times C$,

donde $\Omega \subset D$. Un documento d_j es llamado un ejemplo positivo de c_i si $\Phi(d_j, c_i) = T$, un ejemplo negativo c_i si $\Phi(d_j, c_i) = F$.

Una vez que el clasificador ha sido construido es deseable evaluar su efectividad, para este caso el corpus inicial se divide en dos conjuntos, no necesariamente de igual tamaño:

- **Training (-and-validation) set** $TV = \{d_1, d_{TV}\}$. Este conjunto de documentos observan las características del conjunto de categorías de las cuales los clasificadores fueron construidos.
- **Test set** $Te = \{d_{TV}^{+1}, d_{\Omega}\}$. Este conjunto se usará con el propósito de testear la efectividad de los clasificadores. Cada documento en Te “alimentará” al clasificador y a la decisión $\Phi(d_j, c_i)$ comparada con la decisión del experto $\tilde{\Phi}(d_j, c_i)$; una medida de clasificación de efectividad se basará en la coincidencia entre los valores de $\Phi(d_j, c_i)$ y $\tilde{\Phi}(d_j, c_i)$.

Cabe destacar como observación, que ninguno de los documentos en Te serán ocupados en la construcción del clasificador.

De la última proposición investigada [9], se resaltaré la etapa de representación del texto, que lleva la discusión a tópicos no abordados en las propuestas anteriores:

Un problema fundamental cuando se trabaja con lenguaje natural, es que el contexto tiene una preponderancia importante en el sentido o significado del texto a clasificar. Por ejemplo la frase “se me ha roto la muñeca”, en este caso la palabra muñeca tiene distintos significados dependiendo del contexto, se puede referir al juguete o a la parte del brazo. Es por esto que los distintos enfoques para representar texto para clasificación de texto reconocen o ignoran esta dependencia según su importancia, pudiéndose estructurar acorde al nivel en que se quiere analizar el texto en orden ascendente como sigue:

- A nivel de sub-palabras (Sub-Word Level): descomposición de palabras y su morfología
- A nivel de palabras (Word Level): las palabras en sí e información léxica
- A nivel de multi-palabras (Multi-Word Level): frases e información sintáctica
- A nivel semántico (Semantic Level): el significado del texto
- A nivel pragmático (Pragmatic Level): el significado del texto con respecto a la situación y el contexto

Cada una de estas categorías ha sido considerada útil en el trabajo con lenguaje natural, no obstante no se deben tratar cada una por separado, en cada nivel pueden existir ambigüedades que sólo pueden ser resueltas usando el nivel “superior”.

A continuación se presentarán distintas representaciones basadas en los niveles descritos anteriormente, se comenzará con el Word Level ya que es la manera más común de representar texto para la clasificación de texto y será una de las que se usará en el desarrollo

del proyecto de título y por otro lado se excluirá el nivel pragmático, ya que es un nivel de representación poco usado en la investigaciones recientes.

3.4.1.1 Word level

En muchos casos, las palabras por si solas son unidades significativas para comprender un texto, incluso sin tener en cuenta el contexto, y en el caso de las palabras como muñeca, detallado anteriormente, tienen un mínimo impacto en la representación del texto como un todo. Es por esto que la representación basada a nivel de palabras es considerada muy efectiva en temas como la recuperación de información y la clasificación de texto. Siendo inclusive la base para la mayoría de trabajos realizados en clasificación de texto.

Por otra parte, una de las ventajas más destacadas del uso de esta representación es su simplicidad. Es relativamente sencillo construir algoritmos capaces de descomponer texto y convertirlo en palabras.

Ignorando la estructura lógica del texto, éste se puede convertir de cadenas de caracteres a una secuencia de palabras, además se asume que el orden de las palabras es irrelevante (o al menos de menor importancia) en la clasificación de texto. Entonces se puede proyectar esta secuencia de palabras en una bolsa de palabras, pero en este caso lo que se guarda, es la frecuencia de ésta en el cuerpo del texto, esta representación es comúnmente llamada como el enfoque del **Bag of Words** (BoW), una representación típica se muestra a continuación

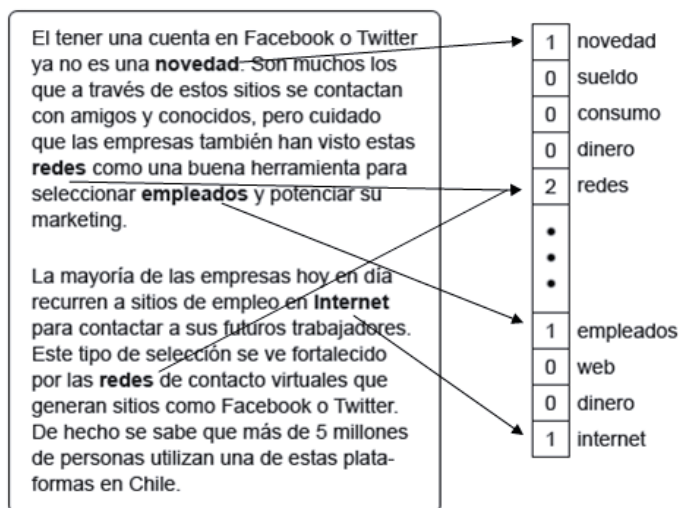


Figura 3.1 Representación típica del BoW

3.4.1.2 Sub-Word level

En vez de usar palabras como índices de términos para la representación de texto, se usan cadenas de n caracteres a partir de las palabras que componen un texto. La representación más popular es el llamado n-grama. Por ejemplo si se usara un trigramma los "bloques" de palabras serían cadenas de 3 caracteres y en el caso de la palabra "bote" su representación como 3-grama sería: "_bo", "bot", "te_", "ote". El texto completo se representaría con una

bolsa de palabras, pero que contengan las del tipo descrito en el ejemplo anterior. Este modelo de representación claramente busca representar la similitud entre palabras, por ejemplo “computador“ y “computadora” no son la misma palabra, pero quieren decir lo mismo, usando esta representación compartirían la mayoría de sus trigramas, como consecuencia es una similitud “correcta”, por contraparte puede suceder lo contrario, la similitud encontrada por el trigramas lleva al error, como sucedería si se representaran las palabras “computador” y “conmutador”, cuyo significado es distinto, además esta representación puede ser efectiva ante errores de ortografía en ciertas palabras, ya que encuentra similitud entre las mismas.

Un resultado de la efectividad del n-grama se muestra en la figura 3.2, donde al comparar un conjunto de palabras usando n-gramas

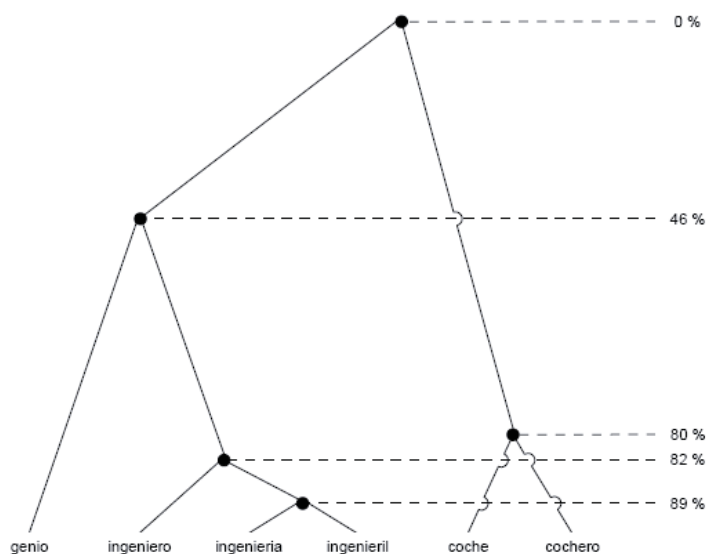


Figura 3.2 Porcentaje de similitud entre palabras usando n-gramas

Se observa en la Figura 3.2 que las palabras que comparten un alto grado de su morfología tienen mayor probabilidad de ser identificadas como similares usando esta representación.

3.4.1.3 Multi-Word level

Con mejoradas herramientas lingüistas, gran cantidad de texto puede ser analizada eficientemente con respecto a su estructura sintáctica, por lo que obviamente las representaciones a este nivel incorporan información sintáctica. Existen dos enfoques que se usan para abarcar este nivel son: basados en sintagmas nominales y los basados en métodos estadísticos.

3.4.1.4 Semantic level

La semántica de un documento puede ser capturada usando taxonomías e indexando lenguajes con un vocabulario fijo. Los documentos son asignados manualmente a una o múltiples categorías y usando una estructura jerárquica de las categorías permiten inferir si existe una relación entre documento y categoría.

Otra técnica es **Latent Semantic Indexing** técnica estadística para estimar la estructura latente. Esta técnica utiliza un valor singular de descomposición que segmenta una gran matriz de datos de asociación de término-documento y permite construir un "espacio semántico" en el que se asocian entre sí términos y documentos. El sustento estadístico en definitiva está determinado por la co-ocurrencia de palabras en la diversidad de documentos.

Revisados todos los puntos tomadas como base, se concluye como propuesta final a dividir el proceso de utilización de máquinas de aprendizaje para la clasificación de texto, en tres etapas las que serán abarcadas en los siguientes tópicos: Representación del Texto (teniendo como base el modelo vectorial de Salton), Clasificadores de Texto (sus distintos tipos y los que se usan más frecuentemente en la clasificación de texto), y una Evaluación de la clasificación las cuales se desarrollarán a continuación.

3.4.2 Representación de texto

Aunque el texto está guardado en un "lenguaje" que puede ser leído por la máquina (html, txt), no es el más adecuado para la mayoría de los algoritmos de aprendizaje. Por consiguiente debe ser transformado de tal forma, que su representación sea la adecuada tanto para los algoritmos de aprendizaje como también para las tareas de clasificación.

Como se ha mencionado, la Recuperación de Información está estrechamente ligada a la clasificación de texto, el punto de partida es similar, por ello el primer problema a enfrentar se refiere a la característica del contenido de los documentos a analizar: documentos en lenguaje natural, surgiendo así el hecho obligatorio de realizar una transformación al documento de manera que las máquinas puedan analizar. Para esto el modelo de espacio vectorial es una de las primeras soluciones realizadas para la representación de documentos y es, hasta la actualidad la más ocupada. Este modelo busca representar documentos en forma de vectores, transformando un documento de texto en un vector que refleje la frecuencia de las palabras que lo componen. Al aplicar este principio sobre un conjunto de documentos, se obtiene un conjunto de vectores formando un espacio vectorial, formado por los vectores representantes de los documentos del conjunto, y cuyas dimensiones serán los términos encontrados.

Para comprender mejor se hace necesario explicar formalmente el modelo de espacio vectorial.

3.4.3 Modelo de espacio vectorial

Usando la definición de [16], el modelo de espacio vectorial intenta recoger la relación de cada documento D_i , de una colección de N documentos, con el conjunto de las m características de la colección. Formalmente, un documento puede considerarse como un vector que expresa la relación del documento con cada una de esas características.

$$D_i \rightarrow \vec{d} = (c_{i1}, c_{i2}, \dots, c_{im})$$

El vector identifica en qué grado el documento D_i satisface cada una de las m características. En otras palabras el vector, c_{ik} es un valor numérico que expresa en qué grado el documento D_i posee la característica k . La "característica" se refiere a la ocurrencia de

determinadas palabras o términos en el documento, o también se pueden tomar en cuenta otros tópicos. Si se consideran las palabras como características definitorias del documento, el proceso que debe seguir el sistema de clasificación se inicia con la selección de aquellas palabras útiles que permitan discriminar unos documentos de otros. En este punto, debemos señalar que no todas las palabras contribuyen con la misma importancia en la caracterización del documento. Desde el punto de vista lingüístico aplicado a la recuperación o clasificación de documentos, existen elementos que no aportan al contenido semántico, como los artículos, las preposiciones o las conjunciones. Estos elementos son conocidos como palabras funcionales en la tradición lingüística y como **stopwords** en el procesamiento de lenguaje natural. Estas palabras, que en español comúnmente son entre 100 y 200, son poco útiles para el proceso de clasificación. También son poco importantes aquellas palabras que por su frecuencia de aparición en toda la colección de documentos pierden su poder de discriminación, es por ello que o son eliminadas o son ponderadas con muy bajo peso estadístico.

Una palabra puede aparecer más de una vez en el mismo documento y, además, algunas palabras pueden considerarse con más peso estadístico que otras, esto es, más significativo que otras, de forma que el valor numérico de cada uno de los componentes del vector obedece normalmente a cálculos más sofisticados que la simple asignación binaria. Por otro lado, también es importante normalizar los vectores para no privilegiar documentos, es por esto que se han propuesto diversos métodos para calcular el peso de cada término en el vector que representa al documento, pero en general, para estimarlos se parte de dos ideas en cierto sentido contrapuestas: si un término se consigna mucho en un documento, aquel es importante para caracterizar el documento. Pero si aparece en muchos documentos de la colección, este término no resulta beneficioso para distinguir un documento de los demás, dado su escaso poder discriminatorio, siendo por tanto poco útil para la recuperación o clasificación de nuevos documentos.

Por lo anterior existe la función de peso $w(D_i, t_j)$, peso del j -ésimo término en el documento i , que busca representar la relación entre un documento y cada término que lo compone.

Suponiendo: f_{ij} la frecuencia del término- j en el documento- i , $|D_i|$ la cantidad de palabras que forman el documento- i , n_j el número de documentos que poseen al término- j , N el número total de documentos, se tienen las siguientes funciones de peso:

Binaria: el valor asignado es un valor binario, e indica la existencia o ausencia del término j en el documento.

$$w(D_i, t_j) = \begin{cases} 1 & \text{si } f_{ij} > 0 \\ 0 & \sim \end{cases}$$

Frecuencia del término: corresponde a la frecuencia absoluta del término dentro del documento.

$$w(D_i, t_j) = f_{ij}$$

TF, Frecuencia normalizada del término: similar a la anterior, sólo que el valor contabilizado es normalizado por el tamaño del documento.

$$w(D_i, t_j) = TF = \frac{f_{ij}}{|D_i|}$$

TF-IDF: La frecuencia normalizada de cada término, ponderado por la frecuencia inversa del término en la colección.

$$w(D_i, t_j) = TF * IDF = \frac{f_{ij}}{|D_i|} * -\log\left(\frac{n_j}{N}\right)$$

TF-RF: Esta es una representación propuesta en [16], es la frecuencia normalizada de cada término, ponderada por la frecuencia de los términos relevantes de la colección.

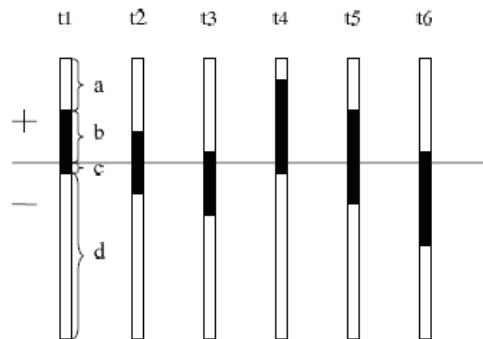


Figura 3.3 Términos Relevantes

En la figura 4.3 los t_i corresponden a los distintos términos de la colección, b y c corresponden a la cantidad de documentos que contienen esos términos con b para los clasificados en la categoría $+$ y c para $-$. Por lo tanto el peso del vector está determinado por:

$$w(D_i, t_j) = TF * rf = \frac{f_{ij}}{|D_i|} * \log\left(2 + \frac{b}{c}\right)$$

El resultado de aplicar esta transformación sobre documentos, será un conjunto de vectores que formarán un espacio vectorial con dimensiones iguales a las palabras encontradas en el total de documentos, lo que conlleva a una alta dimensionalidad del espacio, ocasionando

un problema, que se debe resolver aplicando reducción de la dimensionalidad, tema que se tratará a continuación.

3.4.4 Reducción de la dimensionalidad

Existen tipos de reducción simples que consisten en modificar la representación del texto usando un paso de pre-procesamiento que se realiza antes de aplicar el clasificador, es una práctica común en la clasificación de texto dos de las técnicas más usadas son:

3.4.4.1 Eliminación de palabras comunes (Stopwords)

En cada idioma siempre existirá un conjunto de palabras de uso común, tales como artículos, pronombres o adverbios. Estas son palabras de poca importancia, ya que no aportan mayor significado. Si se construye una lista de estas palabras, se podrán filtrar de la lista de palabras producida por el proceso de indexación. Este proceso busca eliminar la cabeza de la distribución de palabras, donde se ubican todas aquellas palabras de uso común que no aportan significado discriminatorio a los documentos.

3.4.4.2 Reducción a la raíz (Stemming)

Para solucionar el problema de la multiplicidad de formas para un mismo concepto (las diferencias en género, número o forma verbal), se utiliza un método conocido como stemming, el cual busca reducir a su raíz las palabras encontradas. Básicamente consiste en un procedimiento de normalización de palabras, eliminando generalmente las terminaciones de éstas. Una forma de implementar este proceso es mediante algoritmos de stemming (algoritmo de Porter).

Además existen dos tendencias que son más que un mero pre-procesamiento, y nacen a partir de la siguiente problemática, la de remover atributos inapropiados o irrelevantes de la representación de texto, una de las tendencias más seguidas es la selección de características o **Feature Selection**, la cual tiene como objetivos:

- Impedir que la máquina de aprendizaje se sobreentrene, producto del entrenamiento con datos que causen un mal ajuste de la función objetivo.
- Mejorar la eficiencia computacional, ya que mucho de los algoritmos no pueden manejar de una manera óptima un espacio vectorial de alta dimensión.

La selección de características permite dos enfoques, Feature Subset Selection y Feature Construction [15], ésta última consiste en introducir nuevas características que tienen como meta representar la mayor información posible de la representación original mientras se minimiza el número de atributos. Una de las técnicas para aplicar este enfoque es la anteriormente mencionada LSI, es decir trabaja más a un nivel semántico, por lo cual no será descrita, ya que escapa de la idea principal de este proyecto, que pretende trabajar a nivel de palabras.

En cambio la selección de subconjuntos de características (Feature Subset Selection), consiste en una nueva representación compuesta de subconjuntos de la representación original, por lo que no se considera un problema, este enfoque cuenta con una serie de métodos:

Eliminación de Stopwords, Document Frequency Thresholding, Información Mutua y χ^2 -statistic los que se detallarán a continuación:

3.4.4.3 Document Frequency Thresholding

La frecuencia de documentos para una palabra, es el número de documentos en que esa palabra se encuentra, en el caso de esta técnica se calcula esa frecuencia para cada palabra perteneciente al conjunto de entrenamiento y se remueven las palabras cuya frecuencia sea menor a un cierto umbral (threshold) predeterminado, con esto se desea lograr que palabras raras o poco “significativas” no influyan en el rendimiento global del conjunto de entrenamiento. [15]

3.4.4.4 Información Mutua (Mutual Information, Information Gain)

Mide el número de bits de información obtenidos por una predicción de categoría, basándose en la presencia o ausencia de una palabra en un texto.

Supongamos que c_1, \dots, c_k denotan el conjunto posible de categorías, la información mutua (IG) de una palabra w está dada por:

$$IG(w) = - \sum_{j=1}^K P(c_j) \log P(c_j) + P(w) \sum_{j=1}^K P(c_j|w) \log P(c_j|w) + P(\bar{w}) \sum_{j=1}^K P(c_j|\bar{w}) \log P(c_j|\bar{w})$$

Donde $P(c_j)$ puede ser estimada de la fracción de documentos en el conjunto total que pertenece a la clase de categorías c_j , y $P(w)$ se estima de la fracción de documentos en que la palabra w se encuentra, además $P(c_j|w)$ puede ser calculada como la fracción de documentos de la clase c_j que tienen al menos una ocurrencia de la palabra w , $P(c_j|\bar{w})$ se calcula como una fracción de documentos de la clase c_j que no contienen la palabra w .

IG se calcula para cada palabra del conjunto de entrenamiento, y las palabras cuyo IG sea menor a un umbral predeterminado deberán ser removidas del vector.

3.4.4.5 χ^2 -statistic

Mide la falta de independencia entre una palabra w y una clase de categoría c_j . Está dada por

$$\chi^2(w, c_j) = \frac{N \times (AD - CB)^2}{(A + C) \times (B + D) \times (A + B) \times (C + D)}$$

Donde A el número de documentos de la clase c_j que contienen la palabra w , B es el número de documentos que contienen w , pero no pertenecen a la clase c_j , C es el número de documentos de la clase c_j que no contienen la clase w , y D es el número de documentos que pertenecen a la clase c_j que no contiene la palabra w . N es el total de documentos.

3.4.4.6 Similitud entre vectores

Una vez que los textos estén representados por vectores “correctamente” construidos, se hace necesario comparar los nuevos documentos, que serán seleccionados automáticamente,

con los ya clasificados, para eso se hace necesario disponer de representaciones vectoriales homogéneas para ambos tipos de texto y obtener el grado de similitud entre ambos, representados como vectores en un espacio multidimensional. Todo esto con el fin de lograr clasificar el texto entrante.

La comparación entre los vectores debe establecer el grado de semejanza existente entre el vector que representa el texto entrante y el vector que representa a cada uno de los textos de la colección. Para un texto nuevo determinado, cada texto representado en el vector de la colección arrojará un grado de similitud determinado; aquellos cuyo grado de similitud sean más elevados se ajustarán mejor a la categoría correspondiente.

El modo más simple de calcular la similitud entre un texto entrante (representado vectorialmente), con el vector representante de la colección de textos es, utilizar el modelo vectorial, realizando el producto escalar de los vectores que los representan. Se incluye además la normalización de los vectores, con el objetivo de evitar distorsiones producidas por los diferentes tamaños de los textos. El índice de similitud más utilizado es el coseno del ángulo formado por ambos vectores. Cabe destacar, que existen otros métodos propuestos para calcular la similitud, algunos ejemplos son: el coeficiente de Dice y el coeficiente de Jaccard entre otros. Para un texto entrante Q , el índice de similitud con un texto de la colección D está dado para los distintos coeficientes en la figura 3.4

Producto punto	$Sim(D, Q) = \sum(a_i * b_i)$
Coseno	$Sim(D, Q) = \frac{\sum(a_i * b_i)}{\sqrt{\sum a_i^2 * \sum b_i^2}}$
Dice	$Sim(D, Q) = \frac{2 \sum(a_i * b_i)}{\sum a_i^2 + \sum b_i^2}$
Jaccard	$Sim(D, Q) = \frac{\sum(a_i * b_i)}{\sum a_i^2 + \sum b_i^2 - \sum(a_i * b_i)}$

Figura 3.4 Fórmulas para la obtención de la similitud entre vectores

3.4.5 Clasificadores de texto

El problema del proceso inductivo de construcción del clasificador de texto se ha abordado de una variedad de formas, a continuación se detallarán los métodos más populares en la literatura.

3.4.5.1 Clasificadores con reglas de decisión

Consiste en un conjunto de formulas condicionales o “clausulas”. Mediante las premisas de las clausulas se denota la presencia o ausencia de términos en el documento de prueba, mientras la clausula cabecera permite decidir si clasificar o no los términos bajo la

categoría c. Una ventaja sobre los arboles de decisión es que se generan clasificadores mas compactos.

Estos clasificadores verifican estas reglas o clausulas sobre el documento para poder clasificarlo. El algoritmo puede eliminar o fusionar clausulas para hacerlo más compacto sin comprometer su efectividad.

<i>wheat & farm</i>	→	WHEAT
<i>wheat & commodity</i>	→	WHEAT
<i>bushels & export</i>	→	WHEAT
<i>wheat & agriculture</i>	→	WHEAT
<i>wheat & tonnes</i>	→	WHEAT
<i>wheat & winter & ¬ soft</i>	→	WHEAT

Figura 3.5 Reglas o cláusulas [15]

3.4.5.2 Arboles de decisión

Un clasificador de texto de árbol de decisión tiene una estructura de árbol en donde los nodos internos son etiquetados por términos, las ramas de estos están etiquetadas con el peso que ese término tiene en el documento de prueba y las hojas son etiquetadas con las categorías. Este clasificador determina la categoría de un documento probando recursivamente los pesos de los términos que los nodos internos tienen sobre él, hasta llegar a una hoja, la etiqueta de esta hoja es asignada entonces al documento. La mayoría de los clasificadores de árbol son representaciones binarias de los textos, por lo que consisten de arboles binarios. En la figura 3.6 se observa la estructura básica de un árbol y los nodos subrayados significan complemento [15].

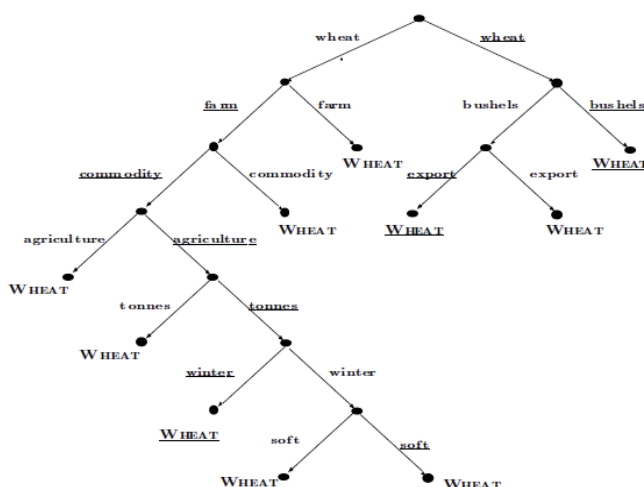


Figura 3.6 Árbol de decisión equivalente a las reglas

3.4.5.3 Métodos en línea

Los métodos en línea (o métodos incrementales) construyen el clasificador tan pronto examinan el primer documento de entrenamiento, y lo refinan incrementalmente a medida que examinan documentos nuevos. Esto tiene una gran ventaja en aplicaciones donde la totalidad

de documentos de entrenamiento no está disponible desde el principio, o en aquellas en las que el sentido de la categoría pueda cambiar en el tiempo, como por ejemplo un filtro adaptativo. También es bastante útil en aplicaciones en las que se espera una realimentación del usuario del clasificador acerca de cómo se están clasificando los textos, así optimizando el proceso con cada ciclo de clasificación y cada “observación” del usuario.

3.4.5.4 Algoritmo de Rocchio

El método Rocchio es la adaptación del algoritmo de Rocchio a la clasificación de texto. El algoritmo de Rocchio proporciona un método para generar los patrones de cada categoría de documentos mediante una fórmula que considera la consulta, el número de documentos relevantes y el número de documentos irrelevantes. Así, con un grupo de datos de entrenamiento previamente clasificados, se aplica el modelo vectorial y se puede construir el patrón para cada clase considerando ejemplos positivos de los documentos de entrenamiento y ejemplos negativos de documentos de otras categorías.

3.4.5.5 Algoritmo del vecino más próximo y variantes

El algoritmo del vecino más próximo (Nearest Neighbour, NN) calcula la similitud entre los documentos a clasificar y cada uno de los documentos de entrenamiento, previamente clasificados, considerando la categoría del más parecido como la categoría a la que pertenece el documento nuevo.

Una de las variantes más conocidas es la del k-nearest neighbour (KNN), que en vez de tomar solo un documento como el más parecido, toma los k documentos más parecidos. Como presumiblemente en estos k documentos se encontrarán varias categorías, se suma la cantidad de documentos de cada categoría y la que más suma será la categoría con la que se clasificará el documento.

El entrenamiento de KNN no es más que la indexación y descripción automática de los documentos, por lo que este proceso y su aplicación son sencillos de aplicar, y es especialmente eficaz cuando el número de categorías es alto y cuando los documentos son heterogéneos y difusos.

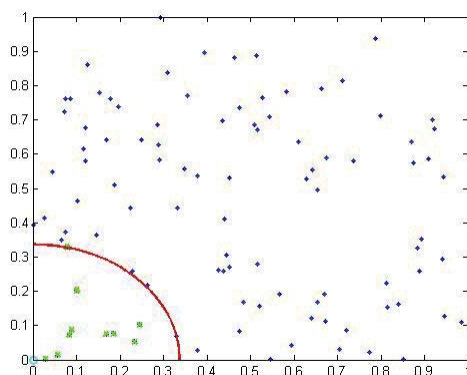


Figura 3.7 Representación gráfica del KNN

3.4.5.6 Redes neuronales

Las redes neuronales constan de varias capas de unidades de procesamiento o neuronal interconectadas. En el caso de la clasificación de texto la capa de entrada recibe términos, mientras que las unidades o neuronas de la capa de salida representan las clases o categorías. Las interconexiones tienen un coeficiente que expresa la mayor o menor fuerza de la conexión, llamados pesos. Es posible entrenar una red para que, con una entrada determinada de los términos de un documento, produzca la salida deseada (la categoría que corresponde al documento). El proceso de entrenamiento de este sistema consiste en ajustar los pesos de las interconexiones, con el fin de que la salida sea la deseada.

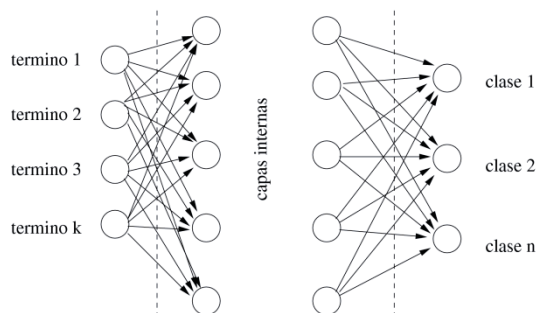


Figura 3.8 Red Neuronal

Se pondrá un especial énfasis en Máquina de Soporte de Vectores (SVM) y Bayes Ingenuo (Naive Bayes), ya que serán los escogidos para este proyecto, como ha sido la tendencia en la decisión de las distintas técnicas a ocupar, una de las razones ha sido su popularidad en la literatura, reflejada en resultados comparativos con el resto de los clasificadores, y por los antecedentes de su uso en documentos en español, estos clasificadores se tratarán a continuación.

3.4.5.7 Naive Bayes

3.4.5.7.1 Introducción al aprendizaje bayesiano

Los métodos de aprendizaje bayesianos son relevantes en el estudio de máquinas de aprendizaje por dos diferentes razones. Primero, los algoritmos de aprendizaje bayesiano que calculan explícitamente probabilidad para hipótesis, como el Naive Bayes, están entre los enfoques más prácticos para cierto tipo de problemas de aprendizaje, por ejemplo, al compararlo con otros algoritmos de aprendizaje como árboles de decisión y redes neuronales se puede comprobar que Naive Bayes es competitivo con respecto a éstos en la mayoría de los casos e inclusive en ciertos casos logra un desempeño mejor. El segundo motivo, tiene relación con el hecho de que al usar métodos Bayesianos en el estudio de máquinas de aprendizaje, proveen una útil perspectiva para comprender muchos de los algoritmos de aprendizaje que no manipulan explícitamente probabilidades, tal es el caso por ejemplo del uso de la perspectiva Bayesiana para analizar el sesgo inductivo de los árboles de decisión, en el diseño de redes neuronales, entre otros [13].

Características de los métodos de aprendizaje bayesiano incluyen:

- Cada ejemplo de entrenamiento (training) observado puede incrementalmente aumentar o disminuir la probabilidad de que una hipótesis sea correcta, esto entrega un enfoque más flexible al aprendizaje comparado con algoritmos que eliminan completamente una hipótesis si se encuentra que es inconsistente con cualquiera de cada uno de los ejemplos.
- El conocimiento previo puede ser combinado con datos observados para así determinar la probabilidad final de una hipótesis. En el aprendizaje Bayesiano, el conocimiento previo está dado por afirmar una probabilidad previa para cada hipótesis candidata y por una distribución probabilística sobre los datos observados para cada hipótesis posible.
- Los métodos bayesianos pueden acomodar hipótesis que hacen predicciones probabilísticas (hipótesis del tipo “Chile tiene un 75% de chances de ser campeón en Sudáfrica”).
- Nuevas instancias pueden ser clasificadas combinando las predicciones de múltiples hipótesis y el peso de sus respectivas probabilidades.
- Inclusive en casos en donde los métodos bayesianos demuestran que éstos no se pueden resolver computacionalmente, pueden proveer un nivel de decisión óptima, que pueden ser usados como medida para otros métodos prácticos.

Por otra parte una de las dificultades prácticas en la aplicación de métodos bayesianos es que en la mayoría de los casos requieren un conocimiento inicial de probabilidades, cuando éstas no son conocidas de antemano son usualmente estimadas basándose en el conocimiento previo, los datos disponibles inicialmente y suposiciones acerca de la forma de las distribuciones de base. Una segunda dificultad práctica es el costo computacional requerido para determinar la hipótesis óptima bayesiana en un caso general.

Una de las aplicaciones de aprendizaje bayesiano en particular es la clasificación de documentos o noticias electrónicas, destacándose por sobre el resto el clasificador Naive Bayes, por lo que se tratará en detalle a continuación, haciendo mención antes a un tema indispensable para el desarrollo de este tema: el Teorema de Bayes.

3.4.5.7.2 Teorema de Bayes

Hablando en términos de máquinas de aprendizaje, el objetivo principal es encontrar la mejor hipótesis de un espacio H , dado el conjunto de entrenamiento (training) ya observado D . Una manera de explicar que se quiere expresar como mejor hipótesis, es decir que se quiere obtener la más probable hipótesis, dado el conjunto de entrenamiento D más cualquier conocimiento inicial acerca de las probabilidades a priori de las diversas hipótesis en H . Para esto Bayes provee un método directo para calcular dichas probabilidades, el **Teorema de Bayes** provee una forma de calcular la probabilidad de una hipótesis basado en: su probabilidad a priori, la probabilidad de los diversos datos observados dados por la hipótesis y los datos observados por sí mismos [13].

Para definir el teorema de Bayes específicamente, es necesario introducir la siguiente notación. $P(h)$ denotará la probabilidad inicial que la hipótesis h posee, antes de observar los datos de entrenamiento. Esta probabilidad es conocida como la probabilidad a priori de h y representa conocimiento base que se tiene acerca de la chance que tiene h de ser una hipótesis correcta. Si no se tiene este conocimiento base, se puede asignar la misma probabilidad a priori a cada una de las hipótesis. Análogamente, se denotará $P(D)$ para representar la probabilidad a priori con la que los datos de entrenamiento D serán observados. Luego, se usará $P(D|h)$ que denota la probabilidad del conjunto de datos observados D dado la condición en que h se sostiene como hipótesis. En problemas de máquinas de aprendizaje se está interesado en la probabilidad $P(h|D)$ que mantiene h dado el conjunto de datos observados D . $P(h|D)$ es llamada la probabilidad a posteriori de h , porque refleja la “confianza” que se mantenga la hipótesis h después de ver los datos de entrenamiento D . Resumiendo la probabilidad a posteriori refleja la influencia del conjunto de entrenamiento D , al contrario de la probabilidad a priori, que es independiente de D .

El Teorema de Bayes es la piedra angular de los métodos de aprendizaje Bayesiano porque provee una manera de calcular la probabilidad a posteriori $P(h|D)$, a partir de la probabilidad $P(h)$, junto a $P(D)$ y $P(D|h)$, como se muestra a continuación:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)} \quad (\text{Ecuación 3.1})$$

Observando la formula se puede deducir que $P(h|D)$ aumenta a medida que $P(h)$ y $P(D|h)$ lo hacen y decrece cuando $P(D)$ se incrementa.

En muchos casos prácticos de máquinas de aprendizaje, el learner considera un conjunto de hipótesis H y teniendo como objetivo encontrar la hipótesis más probable $h \in H$ dado el conjunto observado D (o al menos una de las máximas probables en caso de haber muchas). Cualquier hipótesis de máximo probable es llamada hipótesis MAP (máximo a posteriori). Se puede obtener ésta, usando el teorema de Bayes para calcular la probabilidad a posteriori para cada hipótesis candidata, por lo que h_{MAP} está dado por:

$$h_{MAP} \equiv \operatorname{argmax}_{h \in H} P(h|D) \equiv \operatorname{argmax}_{h \in H} \frac{P(D|h)P(h)}{P(D)} \equiv \operatorname{argmax}_{h \in H} P(D|h)P(h) \quad (\text{Ecuación 3.2})$$

En el último paso se eliminó $P(D)$, porque es una constante independiente de h .

En algunos casos, se asume que cada hipótesis H , es igualmente probable a priori ($P(h_i) = P(h_j)$ para cada h_i y h_j en H). Por lo que la ecuación (3.2) se puede simplificar aún más, para esto se necesita considerar solamente $P(D|h)$ para encontrar la hipótesis más probable. $P(D|h)$ es llamada frecuentemente la verosimilitud del dato D dado h , y cualquiera de las hipótesis que maximiza $P(D|h)$ es llamada hipótesis de máxima verosimilitud, h_{MV} .

$$h_{MV} \equiv \operatorname{argmax}_{h \in H} P(D|h) \quad (\text{Ecuación 3.3})$$

Resumiendo la relación entre el teorema de Bayes y el concepto de aprendizaje es que Bayes provee de principios para calcular la probabilidad a posteriori de cada hipótesis dado el conjunto de entrenamiento, esto se puede usar como la base para un algoritmo aprendizaje

mejor capacitado que calcule la probabilidad para cada hipótesis y de cómo salda la más probable.

Ya definido el Teorema de Bayes corresponde seguir con el otro punto propuesto, el que concierne al algoritmo de aprendizaje Naive Bayes.

3.4.5.7.3 Naive Bayes

El clasificador Naive Bayes se aplica en tareas de aprendizaje, donde cada instancia x es descrita por una conjunción de valores de atributos y con una función objetivo $f(x)$ que puede tomar cualquier valor de un conjunto finito V . Un conjunto de ejemplos de entrenamiento de la función objetivo es provista desde un principio y es posible la llegada de una nueva instancia representada por la tupla de valores de atributo $\langle a_1, a_2 \dots a_n \rangle$. Al learner se le pide predecir el valor objetivo, o dicho de otra manera, clasificar la nueva instancia.

El enfoque bayesiano para clasificar la nueva instancia consiste en asignar el más probable valor objetivo v_{MAP} , dados los valores de atributos $\langle a_1, a_2 \dots a_n \rangle$ que describen la instancia, por lo tanto:

$$v_{MAP} \equiv \operatorname{argmax}_{v_j \in V} P(v_j | a_1, a_2 \dots a_n) \quad (\text{Ecuación 3.4})$$

Usando Bayes, se reescribe la ecuación 3.4 quedando:

$$\begin{aligned} v_{MAP} &\equiv \operatorname{argmax}_{v_j \in V} \frac{P(a_1, a_2 \dots a_n | v_j) P(v_j)}{P(a_1, a_2 \dots a_n)} \\ &\equiv \operatorname{argmax}_{v_j \in V} P(a_1, a_2 \dots a_n | v_j) P(v_j) \end{aligned} \quad (\text{Ecuación 3.5})$$

Es posible estimar los dos términos en la ecuación 5.4 basándose en los datos de entrenamiento. $P(v_j)$ se obtiene fácilmente contando la frecuencia de cada valor objetivo v_j en el conjunto de entrenamiento. Sin embargo estimar los distintos $P(v_j | a_1, a_2 \dots a_n)$ términos de esta manera no es factible a menos que se tenga un conjunto de datos de entrenamiento enorme. El problema es que el número de esos términos es igual al número de posibles instancias multiplicado por el número de posibles valores objetivos.

El clasificador Naive Bayes se basa en la presunción de que los valores de atributos son condicionalmente independientes dado el valor objetivo, es decir la presunción es la siguiente: dado los valores objetivos de la instancia, la probabilidad de observar la conjunción $a_1, a_2 \dots a_n$ es el producto de las probabilidades para los atributos individuales: $P(a_1, a_2 \dots a_n | v_j) = \prod_i P(a_i | v_j)$. Sustituyendo esto en la ecuación 3.5, se tiene la definición para el clasificador Naive Bayes:

$$v_{NB} \equiv \operatorname{argmax}_{v_j \in V} P(v_j) = \prod_i P(a_i | v_j) \quad (\text{Ecuación 3.6})$$

Resumiendo, el método de aprendizaje Naive Bayes involucra un paso de aprendizaje en el cual el conjunto de $P(v_j)$ y $P(a_i/v_j)$ de los términos son estimados, basándose en su frecuencia sobre el conjunto de entrenamiento. Estas estimaciones corresponden a la hipótesis “aprendida”, y es la que se usa al momento de clasificar una nueva instancia aplicando la regla definida en la ecuación 3.6.

En este sentido, con los clasificadores bayesianos ingenuos se asume que el efecto de un valor del atributo en una clase (categoría) dada es independiente de los valores de los otros atributos. Esta suposición se llama "independencia condicional de clase". Ella permite simplificar los cálculos involucrados, siendo por esto que se le considera "ingenuo" (naive) al método y, por lo mismo, sus resultados deben ser entendidos como una simplificación de la realidad. [16].

3.4.5.8 Máquina de vectores de soporte (SVM)

Esta técnica de clasificación de datos es más reciente que la mayoría de clasificadores estudiados (introducida por Vapnick, 1995 y en Categorización de texto por Joachims en 1998), es aplicada en diversas investigaciones logrando un buen desempeño en una amplia variedad de problemas de clasificación, donde se ha destacado en la actualidad, específicamente, en problemas de clasificación de textos, se le reconoce la capacidad de minimizar el error de generalización, es decir, los errores del clasificador a la hora de trabajar sobre nuevos documentos.

Es muy eficaz trabajando con datos multidimensionales, como ocurre al representar textos en forma de vectores, tema que se trató detalladamente en la sección 3.4.3, como se describió anteriormente un proceso común para la clasificación de textos requiere datos para entrenamiento y prueba usando un algoritmo a partir de características cuantificables de los textos. Por lo que cada unidad textual en el grupo de entrenamiento contiene un valor de clasificación, designado por una etiqueta de clase, y múltiples atributos o rasgos. El objetivo de la SVM es producir un modelo que permita predecir los valores de clasificación (identificar la clase) en la etapa de prueba conociendo solo los atributos.

Como características principales las SVM tienen a favor que, se escalan relativamente bien para datos en espacios dimensionales altos; y en contra lo que puede pasar con cualquier algoritmo de aprendizaje la generación de dos problemas el **overtraining**, se han aprendido muy bien los datos de entrenamiento pero no se pueden clasificar bien ejemplos no vistos antes y el **overfitting**, no se ha aprendido muy bien la característica de los datos de entrenamiento, por lo que se hace una mala clasificación.

Se basa en el principio de Minimización del Riesgo Estructural (Vapnick 1982), proveniente de la teoría de aprendizaje estadístico, donde la idea principal es encontrar una hipótesis h desde una hipótesis H para la cual se pueda garantizar la mínima probabilidad de error para una muestra de entrenamiento con n ejemplos.

Y en términos geométricos, lo que busca resolver principalmente la SVM es encontrar una frontera de decisión lineal entre dos grupos o clases (existe también la posibilidad de multiclases), a través de una línea que los separe, maximizando el espacio del hiperplano separador, una muestra gráfica sería la que se describe en la figura 3.9.

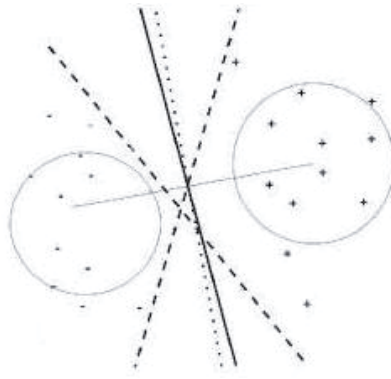


Figura 3.9 Frontera de decisión lineal

Maximizar el margen es el objetivo, ya que los puntos cerca de la superficie de decisión representan decisiones de clasificación inciertas, al tener un buen margen se hace fácil distinguir a que clase corresponde cada dato en el espacio.

Para hallar esos vectores de soporte que maximicen el margen, existen dos métodos fundamentales que se atañen a la naturaleza de los datos, éstos se detallarán a continuación.

3.4.5.8.1 Modelo lineal de vectores de soporte

Para dos clases distintas, existen datos de un conjunto de entrenamiento, que su principal característica radica en que pueden ser linealmente separables, como se muestra en la Figura 3.10

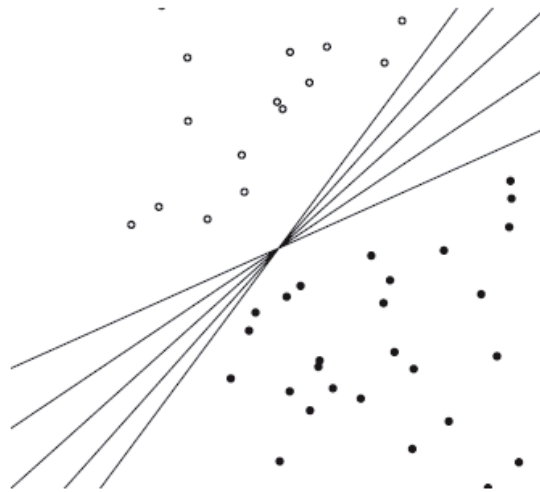


Figura 3.10 Infinitos hiperplanos

Existen infinitos hiperplanos que pueden separar las dos clases existentes (los círculos rellenos y los que no), intuitivamente la decisión para encontrar una frontera, sería dibujar una línea en el medio de la nada entre medio de los datos que representan a ambas clases, para encontrar esa línea existen métodos tales como Bayes Ingenuo, que la encuentra en base a un criterio dado, en el caso de la SVM en particular, define un criterio que busca un hiperplano de separación óptima el cual tiene dos características preponderantes: es único para cada clase de datos separables linealmente, y el riesgo asociado de sobreestimación es más reducido que para cualquier otro hiperplano de separación. La distancia desde ese hiperplano al punto que se encuentre más cercano determinará el **margen** del clasificador y los puntos que determinan el hiperplano de decisión que maximice ese margen corresponden a los **vectores de soporte**, en la figura se ejemplifica la situación descrita anteriormente.

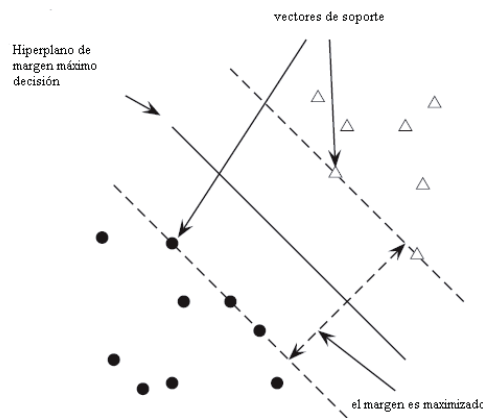


Figura 3.11 Vectores de Soporte

La base teórica con fundamento algebraico es la siguiente:

Dado un set de entrenamiento de instancias de pares (x_i, y_i) , $i = 1 \dots l$ donde $x_i \in R^n$ Y $y_i \in \{1, -1\}^l$ la máquina de vectores de soporte (SVM) busca resolver el siguiente problema de optimización:

$$\begin{aligned} \min_{w, b, \xi} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^l \xi_i \\ \text{subject to} \quad & y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0. \end{aligned}$$

$C > 0$ es el parámetro que penaliza el error en el caso que los datos no sean separables y ξ_i es el límite superior en el número de errores de entrenamiento, mide cuanto (x_i, y_i) se desfasa del margen.

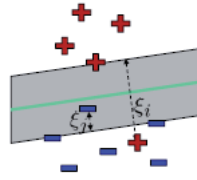


Figura 3.12 Representación visual de errores de entrenamiento

En la mayoría de los casos, ocurre que la búsqueda del hiperplano que maximice el margen resulta ser demasiado restrictiva para un caso práctico, lo que se suele hacer es mapear el espacio de entrada en una dimensión mayor y buscar el hiperplano óptimo allí, entonces sea $z = \varphi(x)$ la notación del vector en el espacio, con un mapeo φ de R^N a un espacio de características Z , se desea encontrar el hiperplano

$$w \cdot z + b = 0$$

Definido por el par (w, b) , tal que se pueda separar el punto x_i de acuerdo a la función

$$f(x_i) = \text{sign}(w \cdot z_i + b) = \begin{cases} 1 & y_i = 1 \\ -1 & y_i = -1 \end{cases}$$

3.4.5.8.2 Modelo no lineal de vectores de soporte

Los clasificadores lineales son inapropiados en muchos problemas de la vida real, ya que los problemas tienen una inherente estructura no-lineal, pero una de las propiedades destacables de las SVM es que pueden transformarse fácilmente a clasificadores no-lineales [9].

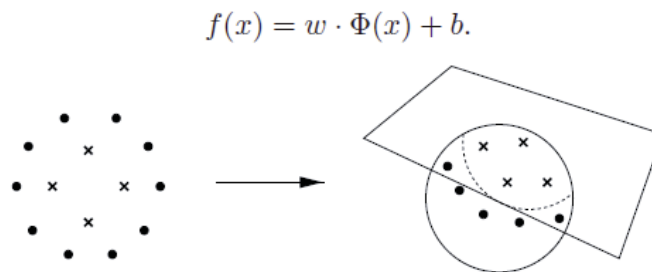


Figura 3.13 Representación visual de mapeo

En principio, el enfoque que se usa es el siguiente, el vector de atributos x_i es mapeado a un espacio de características de mayor dimensión X' usando un mapeo no lineal $\Phi(x_i)$. La

SVM entonces aprende el máximo margen de la regla de clasificación lineal en el espacio de características X' , que es no-lineal cuando se proyecta al espacio de entrada

$$\Phi : R^2 \rightarrow R^3$$

$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

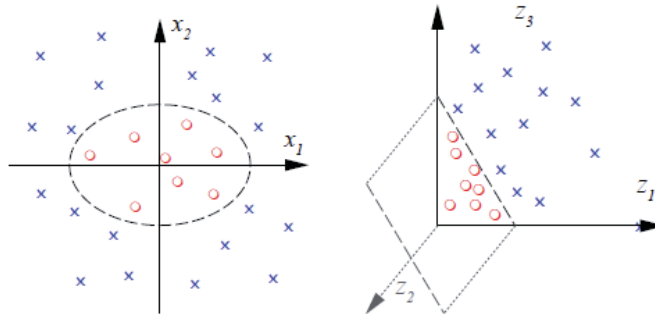


Figura 3.14 Mapeo polinomial

Entonces, cuando los datos no se pueden separar linealmente, se hace un cambio de espacio mediante una función que transforme los datos de manera que se puedan separar. Esa función, se llama Kernel, existen distintas funciones de Kernel. Aunque los más usados, propuestos por los investigadores son básicamente cuatro:

- Kernel Lineal: $K(\vec{a}, \vec{b}) = \vec{a} * \vec{b}$
- Kernel Polinomial de grado d : $K(\vec{a}, \vec{b}) = (\vec{a} * \vec{b} + 1)^d$
- Kernel de Funciones de Radio Basal (RBF): $K(\vec{a}, \vec{b}) = \exp(-\gamma [\vec{a} - \vec{b}]^2)$
- Kernel Sigmoidal: $K(\vec{a}, \vec{b}) = \tanh(\gamma [\vec{a} * \vec{b}] + c)$

Acá γ , d y c son parámetros de los kernel que pueden ajustarse para mejorar el desempeño de la clasificación.

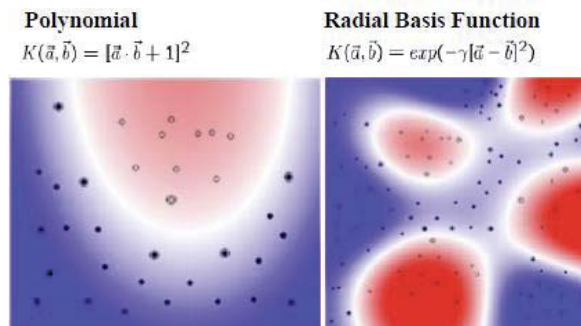


Figura 3.15 Kernel Polinomial y RBF

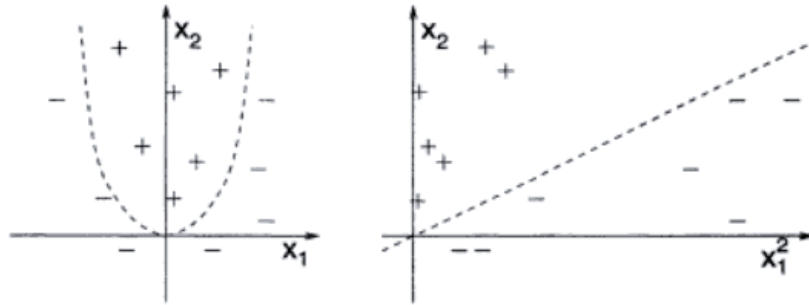


Figura 3.16 Truco del Kernel

Clasificación multiclase

Si bien las máquinas de soporte vectorial fueron diseñadas originalmente para la clasificación binaria. Existen distintos métodos para abordar la clasificación en más de dos clases, estos métodos se pueden separar en dos aproximaciones. Una es construir un clasificador multiclase combinando varios clasificadores binarios y la otra es considerar directamente todos los datos para formular el problema de optimización. A continuación se explicará brevemente los métodos que usan SVM binario como base.

El método **“uno contra todos”** fue el primero en ser usado y de los más fácil, este método construye k clasificadores SVM donde k es el número de clases. El clasificador i tiene como salida la función ρ_i , es entrenado asignando como etiqueta positiva todos los ejemplos de la clase i y todos los otros con etiqueta negativa. Para clasificar nuevos datos se asigna a la clase que contenga el mayor valor de ρ_i .

Otro método importante es el llamado **“uno contra uno”**, en donde se construyen $k(k-1)/2$ clasificadores SVM binarios, todas las combinaciones posibles entre 2 clases, en los que se clasifica entre dos clases distintas. Para un nuevo vector x a probar si el clasificador C_{ij} decide que x pertenece a una clase se indica con un voto para esta clase, sino se incrementa el voto a la otra clase, finalmente se asigna el vector x a la clase con mayor cantidad de votos.

El tercer método que usa SVM binario como base es el método **DAGSVM** (Direct Acyclic Graph). La fase de entrenamiento es la misma que en el método de “uno contra uno” entrenando $k*(k-1)/2$ SVM binarios. Sin embargo en la fase de prueba usa un grafo binario acíclico dirigido con $k*(k-1)/2$ nodos internos y k hojas. Cada nodo es un SVM binario. Dado un vector de prueba, recorre el grafo partiendo del nodo raíz y se mueve a la derecha o izquierda dependiendo del valor de salida del nodo tras ser evaluado. De esta manera sigue el camino hasta llegar el nodo hoja que indica la clase que se asignará.

En el estudio comparativo realizado [3] entre estos métodos, y otros donde se usan todos los datos, se comprobó que en general el método “uno contra uno” es el que presenta mejores tasas de precisión (accuracy) y menores tiempos de entrenamiento, seguido del método de DAGSVM.

Problem	One-against-one		DAG		One-against-all		[25], [27]		C&S	
	<i>C</i>	rate	<i>C</i>	rate	<i>C</i>	rate	<i>C</i>	rate	<i>C</i>	rate
iris	2 ⁴	97.333	2 ⁸	97.333	2 ¹²	96.000	2 ⁵	97.333	2 ⁰	87.333
wine	2 ⁻²	99.438	2 ⁻²	98.315	2 ²	98.876	2 ⁻¹	98.876	2 ⁻¹	99.438
glass	2 ⁸	66.355	2 ⁴	63.551	2 ⁵	58.879	2 ⁹	65.421	2 ⁶	62.617
vowel	2 ⁵	82.954	2 ⁶	81.439	2 ¹¹	50.000	2 ⁸	67.424	2 ⁶	63.068
vehicle	2 ⁵	80.615	2 ⁵	80.851	2 ¹²	78.132	2 ¹⁰	80.142	2 ⁴	79.669
segment	2 ¹²	96.017	2 ¹¹	95.844	2 ¹²	93.160	2 ⁸	95.454	2 ⁻²	92.165

Figura 3.17 Comparación de métodos usando kernel lineal [4]

3.4.6 Evaluación de los clasificadores

3.4.6.1 Introducción

Al igual que en el caso de los sistemas de Recuperación de Información (IR), la evaluación de los clasificadores de documentos, es comúnmente llevada a cabo experimentalmente, más que analíticamente. La razón para esto, es que para evaluar un sistema analíticamente, se debe realizar una especificación formal del problema que el sistema debe resolver, y la idea principal de la clasificación de texto es, debido a su carácter subjetivo, inherentemente no formalizable [15].

La evaluación experimental de un clasificador usualmente mide la efectividad, que para este caso particular corresponde a la habilidad de tomar la decisión de clasificación correcta.

3.4.6.2 Medidas para la efectividad en categorización de texto

3.4.6.2.1 Precision y Recall

La efectividad de los clasificadores en la mayoría de los casos es medida en términos de las clásicas nociones de Recuperación de Información, como son Precision (π) y Recall (ρ), pero adaptadas al caso de categorización de texto. π está definida como la probabilidad condicional descrita a continuación:

$$P(\sim\Phi(d_x, c_i) = T | \Phi(d_x, c_i) = T)$$

La probabilidad de que si un documento al azar d_x es clasificado bajo c_i , la decisión es correcta. Análogamente, ρ está definido por:

$$P(\Phi(d_x, c_i) = T | \sim\Phi(d_x, c_i) = T)$$

La probabilidad de que si un documento al azar d_x es debería ser clasificado bajo c_i , la decisión es tomada. Estos valores relativos a una categoría pueden ser promediados de manera de obtener el valor global de π y ρ para todo el conjunto de categorías, tomando términos de lógica, π viene a representar el grado de “ruido” con el cual el clasificador asigna la categoría C , mientras que ρ puede ser visto como el grado de “integridad” para la decisión tomada por el clasificador.

Estos valores deben ser entendidos como probabilidades subjetivas, ya que miden la expectativa del usuario con respecto a si el sistema se comportará correctamente al momento de clasificar un documento cualquiera bajo una categoría c_i . Estas probabilidades pueden ser estimadas en términos de una tabla de contingencia para la categoría c_i en un conjunto de prueba (test) dado, donde, FP_i (falsos positivos) es el número de documentos del conjunto de entrenamiento que han sido incorrectamente clasificados bajo c_i ; TN_i (verdaderos negativo), TP_i (falsos negativos), FN_i (falsos negativo) se ajustan a la misma definición dependiendo el caso.

Category c_i		expert judgments	
		YES	NO
classifier judgments	YES	TP_i	FP_i
	NO	FN_i	TN_i

Figura 3.18 Tabla de Contingencia para i categorías [15]

Por lo tanto π y ρ usando los términos definidos anteriormente pueden ser estimados entonces por:

$$\pi_i = \frac{TP_i}{TP_i + FP_i} \quad (\text{Ecuación 3.8})$$

$$\rho_i = \frac{TP_i}{TP_i + FN_i} \quad (\text{Ecuación 3.9})$$

3.4.6.2.2 Microaveraging

Está relacionada con la medida anterior, y define que π y ρ se obtienen sumando todo el conjunto de decisiones individuales.

$$\pi = \frac{TP}{TP + FP} = \frac{\sum_{i=1}^{|C|} TP_i}{\sum_{i=1}^{|C|} (TP_i + FP_i)}$$

$$\rho = \frac{TP}{TP + FN} = \frac{\sum_{i=1}^{|C|} TP_i}{\sum_{i=1}^{|C|} (TP_i + FN_i)}$$

3.4.6.2.3 Macroaveraging

Consiste en que π y ρ son primero evaluadas “localmente” (ecuaciones 3.8 y 3.9) para cada categoría, y luego “globalmente” promediando el resultado sobre el número total de las diferentes categorías.

$$\pi = \frac{\sum_{i=1}^{|C|} \pi_i}{|C|}$$

$$\rho = \frac{\sum_{i=1}^{|\mathcal{C}|} \rho_i}{|\mathcal{C}|}$$

La tabla “global” de contingencia que se obtiene sumando todas los valores obtenidos para cada categoría específica y se obtiene un promedio dividiendo por el número total de categorías \mathcal{C} , como se muestra a continuación en la figura 3.19

Category set $\mathcal{C} = \{c_1, \dots, c_{ \mathcal{C} }\}$		expert judgments	
		YES	NO
classifier	YES	$TP = \sum_{i=1}^{ \mathcal{C} } TP_i$	$FP = \sum_{i=1}^{ \mathcal{C} } FP_i$
	NO	$FN = \sum_{i=1}^{ \mathcal{C} } FN_i$	$TN = \sum_{i=1}^{ \mathcal{C} } TN_i$

Figura 3.19 Tabla de contingencia global para todo el conjunto de categorías [15]

3.4.6.2.4 Precision/Recall Breakeven

Dado un ranking de documentos, el punto de Precision/Recall Breakeven es el valor en el cual precisión y recall son iguales.

3.4.6.2.5 F_β -Measure

Precision y Recall describen la performance de la clasificación desde el punto de vista del accuracy, pero al considerarlos como valores separados dificulta la tarea de comparar clasificadores, haciendo imprescindible obtener una única medida de performance, es por ello que se usa una media armónica entre Precision y Recall más conocida como F_β -Measure que se estima de la siguiente manera:

$$F_\beta = \frac{(1+\beta^2)TP}{(1+\beta^2)TP+FP}$$

4 Pruebas sobre algoritmos escogidos

4.1 Introducción

Una forma de evaluar existente es comparar resultados con distintos autores que utilicen distintas metodologías, clasificadores, etc. es decir realizar un benchmark conjunto para la categorización de texto o compararse con algunos ya existentes.

Para propósitos de experimentación, existen colecciones estándar de benchmark, que pueden ser utilizados como conjunto de datos inicial y que están disponibles públicamente. Las más usadas en tareas de categorización de texto (CT) son la colección Reuters, WebKB y Ohsumed [10], todas en el idioma inglés, y que consisten en una serie de conjuntos de datos de entrenamiento. La mayoría de experimentos relacionados con CT a la fecha usan estos conjuntos de datos, pero esto no quiere decir que se traduzca en resultados comparativos confiables, debido a que esos distintos experimentos han sido realizados bajo diferentes condiciones experimentales (distintos procedimientos, etapas, métodos, etc.). Por lo tanto para que los resultados de la experimentación con distintos clasificadores sea directamente comparable, el experimento debe desarrollarse bajo las siguientes condiciones:

- Todos los clasificadores deben ser experimentados bajo la misma colección de datos.
- La división del conjunto total de datos en conjuntos de training y test deben ser la misma para todos los clasificadores.
- La misma medida de efectividad debe ser usada para todos los clasificadores, y si esta medida depende de algún parámetro, la elección de dicho parámetro debe ser la misma para todos los clasificadores también.

A continuación se describirán las colecciones más usadas en tareas de categorización de texto.

4.2 Colecciones de prueba

4.2.1 Reuters-21578¹

Compilada por David Lewis y originalmente recolectada por el grupo de Carnegie para la edición de Reuters 1987, consiste en un corpus de 9.603 documentos de entrenamiento y 3.299 documentos de prueba. De las 135 potenciales categorías de tópicos sólo 90 de éstas se da el caso que existe al menos un documento de entrenamiento y prueba. Esta colección es conocida por tener un vocabulario bastante restringido, los elementos de entrenamiento cuentan con sólo 27.658 términos distintos que se encuentran al menos una vez, otra característica es que hay una directa correspondencia entre palabras y categorías, por ejemplo para la categoría “Wheat”, la ocurrencia de la palabra wheat en el documento es un buen predictor.

¹ Disponible en <http://www.daviddlewis.com/resources/testcollections/reuters21578/>

4.2.2 WebKB²

Consiste en una colección de páginas WWW habilitadas por el grupo de text-learning de la Carnegie Mellon University, establecen 4 categorías course, faculty, Project y student. Los documentos que no se encuentren ninguna de las 4 categorías son eliminados, arrojando entonces la cantidad de 4.183 documentos, 229 para entrenamiento y 3957 para pruebas. Una característica es que el tipo de texto es muy diferente al de la colección de Reuters, al ser páginas WWW cuentan con estilos de escritura heterogéneos, oraciones incompletas e información estructural. Después de remover el código html, el total de palabras cuenta con 38.359 elementos. Otra característica es que la clasificación no se hace por concordancia al tópico sino por la función que cumple la página.

4.2.3 Ohsumed³

Colección compilada por William Hersh, relativo a temas médicos, de los 50.216 documentos a la fecha en que se creó la colección (1991), se usaron los primeros 10.000 como entrenamiento y los siguientes 10.000 como prueba, cuenta con 38.679 y un vocabulario dominado por términos relativos a la medicina.

Finalmente se eligió la colección de Reuters, por ser la más citada en la literatura y la que más se asemeja en forma (documentos noticiosos) al caso de estudio.

4.3 Caso de estudio

4.3.1 Introducción

El exponencial crecimiento de la internet, conlleva un crecimiento también en el contenido disponible, un caso donde esta situación se hace evidente es la publicación de noticias electrónicas en donde cada medio tradicional se vio prácticamente en la obligación de publicar sus artículos en la internet, y esto se suma además a la nueva tendencia acaecida en los últimos años de los blogs, en donde personas pertenecientes al ámbito periodístico como los que no son libres de publicar artículos relacionadas a las noticias contingentes.

Por lo que para una persona interesada en algún tópico en general se le puede tornar una tarea tediosa el seleccionar manualmente las noticias que desea obtener, es por esto que se hace necesario poseer una herramienta que automatice esta tarea.

Es por esto que se escogió resolver esta problemática como caso de estudio –teniendo como característica que será aplicada en noticias en español- para aplicar métodos de máquinas de aprendizaje en la clasificación de texto, y así poder hacer pruebas sobre los métodos escogidos para la realización de este proyecto.

4.3.2 Obtención del conjunto de datos para la clasificación

Como se mencionó en la introducción al caso de estudio, se utilizarán como conjunto de datos noticias en español específicamente de un medio chileno, La Tercera, para la obtención de éstos se usará un recurso conocido como canal (feed), que posee un archivo RSS,

² Disponible en <http://www.cs.cmu.edu/afs/cs/project/theo-20/www/data/>

³ Disponible en <http://ir.ohsu.edu/ohsumed/>

que es el principal formato para la difusión de contenido en la web (existen otros formatos como ATOM o RDF).

Usando el enfoque propuesto por [9] se necesitarán un conjunto de datos para entrenamiento (training) y para pruebas (test), como es fundamental contar con elementos previamente clasificados para el entrenamiento, se ocuparán 1000 artículos clasificados por La Tercera, 500 de la categoría Deportes y 500 de otras categorías, para el conjunto de pruebas se ocuparán 250 de Deportes y 250 de otras categorías. Se han elegido 1000 artículos ya que en el estudio realizado, se aprecia que al aumentar en 1000 la efectividad en la clasificación no es porcentualmente significativa

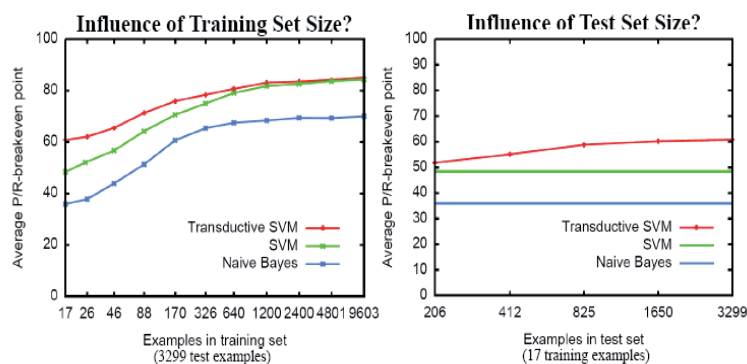


Figura 4.1 Influencia del tamaño en los conjuntos de training y test [8]

Por lo tanto se usarán los canales diferenciados-Deportes, País, Mundo y Negocio en este caso- que ofrece La Tercera en su página web <http://www.latercera.com>.



Figura 4.2 Enmarcado en cuadro rojo el acceso a los distintos canales

Este archivo RSS, está escrito en lenguaje XML-metalenguaje extensible de etiquetas desarrollado por la World Wide Web Consortium (W3C)- permitiendo su manejo de una

forma más sencilla a través de librerías del lenguaje PHP, escogido para la obtención de estos artículos.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<rss xmlns:content="http://purl.org/rss/1.0/modules/content/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:taxo=""
<channel>
<title>latercera.com: Pais</title>
<link>http://latercera.com/canal/654.html</link>
<description>Noticias de latercera.com: Pais</description>
<image>
<title>latercera.com</title>
<url>http://resource.latercera.com/css/i/logo.jpg</url>
<link>http://latercera.com</link>
<description>latercera.com en la Web</description>
</image>
<item>
<title>Hija de Pilar Pérez presenta recurso al TC por cancelación de juicio contra su madre</title>
<link>http://latercera.com/contenido/654_201487_9.shtml</link>
<description><![CDATA[<div align="left" border="0" src="http://static.latercera.com/200911/576984_100.jpg" alt=""
<pubDate>Mon, 16 Nov 2009 23:23:00</pubDate>
<guid>http://latercera.com/contenido/654_201487_9.shtml</guid>
<dc:date>2009-11-16T23:23:00-04:00</dc:date>
</item>
<item>
<title>Detienen a peruano por abuso sexual de menores en Santiago</title>
<link>http://latercera.com/contenido/654_201486_9.shtml</link>
<description>Sujeto cometía bajo amenaza las agresiones contra una adolescente de 17 años. Desde 2007 era buscado por
<pubDate>Mon, 16 Nov 2009 23:16:29</pubDate>
<guid>http://latercera.com/contenido/654_201486_9.shtml</guid>
<dc:date>2009-11-16T23:16:29-04:00</dc:date>
</item>
<item>
<title>Candidatos enfrentan posturas en último debate presidencial</title>
<link>http://latercera.com/contenido/654_201485_9.shtml</link>
<description><![CDATA[<div align="left" border="0" src="http://static.latercera.com/200911/583264_100.jpg" alt=""
<pubDate>Mon, 16 Nov 2009 22:49:00</pubDate>
<guid>http://latercera.com/contenido/654_201485_9.shtml</guid>
<dc:date>2009-11-16T22:49:00-04:00</dc:date>
</item>
</item>
```

Figura 4.3 Código fuente de RSS de la tercera en XML

Una vez obtenidos serán almacenados en una Base de Datos provista por el sistema de gestión de Base de Datos MYSQL. Para esto en primer lugar se debe tener las direcciones de cada canal para acceder a ellos mediante un script PHP.

Mostrar: 30 filas empezando de 0
 en modo horizontal y repetir los encabezados cada 100 celdas
 Organizar según la clave: Ninguna Continuar

	urlblog	urlrss	categoria	pais	nombre	tipofeed
<input type="checkbox"/>	http://www.latercera.com/	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&category=654	Pais	Chile	La Tercera	rss
<input type="checkbox"/>	http://www.latercera.com/	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&category=678	Mundo	Chile	La Tercera	rss
<input type="checkbox"/>	http://www.latercera.com/	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&category=655	Negocios	Chile	La Tercera	rss
<input type="checkbox"/>	http://www.latercera.com/	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&category=656	Deportes	Chile	La Tercera	rss

Mostrar: 30 filas empezando de 0
 en modo horizontal y repetir los encabezados cada 100 celdas

Figura 4.4 Direcciones de los canales almacenados en la Base de Datos

El script anterior se encarga de revisar el archivo RSS y obtener los artículos publicados, para luego almacenarlos en la Base de datos para un posterior procesamiento. Con la observación que el contenido de los artículos queda por default vacío, ya que no se publica

completo en el RSS, mediante la creación de otro script PHP se obtiene y además se convierte el texto a minúsculas, se extraen tildes y signos de puntuación, para efectos de redundancia innecesaria de palabras, estos artículos se irán guardando en tablas diferentes una para training y otra para test, con la salvedad obviamente que no se repitan artículos en las propias tablas como entre ellas.

	urlrss	urlpost	titulo	categoria	contenido	fecha
<input type="checkbox"/>	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&c...	http://latercera.com/contenido/654_200592_9.shtml	Ministra peruana: compra de armamento de Chile es.	Pais	vacio	2009-11-13 18:15:59
<input type="checkbox"/>	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&c...	http://latercera.com/contenido/654_197993_9.shtml	Comuneros presentan denuncia contra Carabineros p...	Pais	representantes de comunidades mapuches presentaron...	2009-11-04 19:04:54
<input type="checkbox"/>	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&c...	http://latercera.com/contenido/654_197983_9.shtml	Explosión deja tres heridos en mercado de Angelmó	Pais	tres personas heridas de distinta consideración fu...	2009-11-04 19:04:54
<input type="checkbox"/>	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&c...	http://latercera.com/contenido/654_197969_9.shtml	Corte desecha último recurso presentado por guarda...	Pais	la septima sala de la corte de apelaciones de sant...	2009-11-04 19:04:54
<input type="checkbox"/>	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&c...	http://latercera.com/contenido/654_197956_9.shtml	Fiscalía indaga difusión de video pornográfico de ...	Pais	la fiscalia de san antonio indagara la difusion de...	2009-11-04 19:04:54
<input type="checkbox"/>	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&c...	http://latercera.com/contenido/654_197951_9.shtml	Delincuentes roban y chocan camión repartidor en S...	Pais	incrustado en un árbol termino esta tarde un camio...	2009-11-04 19:04:54
<input type="checkbox"/>	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&c...	http://latercera.com/contenido/654_197950_9.shtml	Comando de Piñera se reúne con canciller por denun...	Pais	durante una reunion con el ministro mariano fernan...	2009-11-04 19:04:54
<input type="checkbox"/>	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&c...	http://latercera.com/contenido/654_197935_9.shtml	Diputados rechazan proyecto que prohíbe a conserva...	Pais	la camara de diputados rechazo el proyecto de ley ...	2009-11-04 19:04:54
<input type="checkbox"/>	http://www.latercera.com/app/rss?sc=TEFURVJDRVJB&c...	http://latercera.com/contenido/654_197920_9.shtml	Denuncian que recursos destinados a Fundación Frei...	Pais	el diputado udi claudio alvarado denuncio ante la ...	2009-11-04 19:04:54

Figura 4.5 Artículos almacenados en la Base de Datos

4.3.3 Representación de texto de los artículos obtenidos

Ya obtenidos y almacenados los artículos, el siguiente paso es aplicar técnicas de representación de texto (lenguaje natural) para el uso de máquinas de aprendizaje, para el caso de estudio se aplicaron las técnicas descritas anteriormente en este informe, Stopwords para Feature Selection y Frecuencia, TF y TD-IDF como representación del modelo vectorial. El almacenamiento de esta representación será en un archivo de texto plano y en un formato tal que pueda ser utilizado por la herramienta LibSVM, que se empleará para realizar el proceso de clasificación, y que se detallará a continuación. Una vez “limpio” el conjunto de training, el proceso siguiente es crear el vector con todos los features o palabras para almacenarlos en la base de datos y además crear un archivo de texto que será usado posteriormente para crear el BoW para cada artículo.

Como se mencionó anteriormente el paso a continuación es crear el BoW para cada artículo tanto en el conjunto de training como de test y convertirlo al formato necesario para su ejecución con LIBSVM.

→ T ←	uripost	título	categoria	contenido	representacion
<input type="checkbox"/>	http://latercera.com/contenido/654_200592_9.shtml	Ministra peruana: compra de armamento de Chile es "delicado"	Pais	hasta el momento esta confirmada la visita como hoy dia esta confirmada la reunion del presidente garcia con la presidenta michelle bachelet no he recibido instruccion alguna de cambiar los planes hasta ahora dijo al diario el comercio	NULL
<input type="checkbox"/>	http://latercera.com/contenido/654_200613_9.shtml	Más de ocho mil jóvenes participarán de la vigésimo sexta peregrinación al santuario de Yumbel	Pais	la peregrinacion culminara con una eucaristia que sera presidida por el arzobispo de concepcion monseñor ricardo ezzati	NULL
<input type="checkbox"/>	http://latercera.com/contenido/654_200616_9.shtml	PDI detiene por tráfico de drogas y cohecho a ex carabinero en Padre Hurtado	Pais	se informo que el imputado integro las filas de carabineros entre 1984 y 1987 y que ahora se desempeñaba como investigador privado el imputado sera formalizado por los dos delitos antes mencionados	NULL
<input type="checkbox"/>	http://latercera.com/contenido/654_200617_9.shtml	Lobos llama al Minsal a "acatar fallo" que favorece a enferma de cáncer	Pais	insistio en la necesidad de establecer un seguro adicional para las enfermedades de alto costo que proteja adecuadamente a todas las personas sean afiliados de fonasa e isapres para que casos tan dramaticos como este no sigan repitiendose	NULL
<input type="checkbox"/>	http://latercera.com/contenido/654_200585_9.shtml	Campaña parlamentaria abre los fuegos de la franja electoral	Pais	en la franja presidencial los tiempos son los mismos para cada postulante a la moneda sebastian piñera eduardo frei marco enriquez-ominami y jorge arrate cuentan con cinco minutos exactos para exponer sus clips promocionales los que seran transmitidos a las 2040 horas	NULL
<input type="checkbox"/>	http://latercera.com/contenido/655_197202_9.shtml	Fiat frenará producción de marcas de Chrysler y potenciará vehículos eficientes	Negocios	baio el antiguo dueño de chrysler cerberus capital management la automotriz habia suspendido gran parte de su trabajo de desarrollo de vehiculos para conservar efectivo	NULL

Figura 4.6 Artículos en minúsculas, sin tildes ni signos de puntuación

uripost	título	categoria	contenido	representacion
http://latercera.com/contenido/654_200592_9.shtml	Ministra peruana: compra de armamento de Chile es "delicado"	Pais	momento confirmada visita hoy dia confirmada reunion presidente garcia presidenta michelle bachelet he recibido instruccion cambiar planes ahora dijo diario comercio	-1 868:1 1489:1 1822:1 2276:1 2435:2 3176:1 3178:1 3180:1 3219:1 3221:1 4379:1 4595:1 4716:1 5958:1 6061:1 6713:1 6991:1 7204:1 7205:1 7629:1 7944:1 9342:1 9344:1 9346:1 9501:1
http://latercera.com/contenido/654_200613_9.shtml	Más de ocho mil jóvenes participarán de la vigésimo sexta peregrinación al santuario de Yumbel	Pais	peregrinacion culminara eucaristia sera presidida arzobispo concepcion monseñor ricardo ezzati	-1 2382:1 6078:1 7980:1 8289:1 9501:1
http://latercera.com/contenido/654_200616_9.shtml	PDI detiene por tráfico de drogas y cohecho a ex carabinero en Padre Hurtado	Pais	informo imputado integro filas carabineros 1984 1987 ahora desempeñaba investigador privado imputado sera formalizado delitos mencionados	-1 256:1 868:1 1895:1 2969:1 3084:1 4232:1 4836:2 4944:1 4992:1 5614:1 6713:1 8289:1 9501:1
http://latercera.com/contenido/654_200617_9.shtml	Lobos llama al Minsal a "acatar fallo" que favorece a enferma de cáncer	Pais	insistio necesidad establecer seguro adicional enfermedades alto costo proteja adecuadamente personas sean afiliados fonasa isapres casos dramaticos sigan repitiendose	-1 768:1 980:1 982:1 1939:1 2669:1 3806:1 5005:1 6908:1 8198:1 8233:1 9501:1
http://latercera.com/contenido/654_200585_9.shtml	Campaña parlamentaria abre los fuegos de la franja electoral	Pais	franja presidencial tiempos son postulante moneda sebastian piñera eduardo frei marco enriquez-ominami jorge arrate cuentan cinco minutos exactos exponer clips promocionales seran transmitidos 2040 horas	-1 1263:1 1265:1 2115:1 2762:1 3438:1 3623:1 3824:1 4273:1 4282:1 4702:1 5230:1 5749:1 6023:1 6025:1 6067:1 6089:1 6864:1 7203:1 8199:1 8290:1 8469:1 8818:1 8820:1 8957:1 9501:1
http://latercera.com/contenido/655_197202_9.shtml	Fiat frenará producción de marcas de Chrysler y potenciará vehículos eficientes	Negocios	antiguo dueño chrysler cerberus capital management automotriz habia suspendido gran parte desarrollo vehiculos conservar efectivo	-1 1125:1 1424:1 1886:1 2091:1 3050:1 3394:1 3443:1 4500:1 4547:1 5662:1 6693:1 6695:1 8661:1 9201:1 9501:1

Figura 4.7 Artículos sin stopwords y con la representación del BoW en "formato" LIBSVM

Como se mencionó, los archivos llevan la siguiente estructura para ser utilizados en compatible con el formato SVM-Light/LIBSVM:

<clase> <feature>:<valor> <feature>:< valor > ... <feature>:< valor >

Cada línea describe un texto ya clasificado representando la clase al principio con un 1 o -1 (caso binario), seguido de pares feature / valor por cada característica (feature) no nula (de valor cero), separados por un espacio vacío. La línea debe terminar con el carácter '\n'.

Una línea de ejemplo sería:

1 27:1 29:1 269:1 2050:1 2834:1 4525:1 5635:1 9501:1

En donde se representa un texto de la clase positiva (1) con una aparición de las características (palabras) 27, 29, 269, 2050, 2834, 4525, 5635 y 9501 en relación al orden de la lista de palabras.

```

1 27:1 29:1 269:1 2050:1 2834:1 4525:1 5635:1 9501:1
1 55:1 57:1 134:1 137:1 179:1 195:1 544:1 546:1 548:1 589:1 599:1 665:1 872:1 1013:1 1064:1 1187:1 1344:1
1 298:1 423:1 455:1 461:1 465:1 502:1 509:2 511:2 513:2 555:1 614:1 983:1 1192:1 1380:1 1556:2 1975:1 2070:3 207:
1 58:2 173:1 295:1 316:1 440:1 496:1 507:1 509:1 511:1 513:1 609:1 676:1 868:1 891:1 1240:3 1380:1 1671:1 1697:1
1 905:1 1563:1 1654:1 1680:1 2242:1 3373:1 3494:1 3923:1 4363:1 4463:1 4464:1 4466:1 4630:1 5321:1 5400:1 5446:1
1 1528:1 1822:1 2487:1 2744:1 2747:1 3207:1 4012:1 4084:1 4086:1 4525:1 5055:1 5057:1 5170:1 5268:1 5270:1 5400:
1 291:1 759:1 1862:2 1864:2 3120:1 3225:2 3817:1 4135:1 4205:1 5248:1 5390:1 5723:1 5725:1 5727:1 5827:1 6834:1
1 418:1 1679:1 1757:1 1922:1 3030:1 3679:1 3681:1 3683:1 4375:1 4765:1 4950:1 5711:1 6808:1 6832:1 7960:1 7995:1
1 273:1 1239:1 1347:1 1875:1 2190:1 2192:1 2428:1 2794:1 2940:1 3296:1 3600:1 4303:1 4561:1 6054:1 6092:1 6093:1
1 401:1 1917:1 3120:1 3751:1 3753:1 3895:1 4111:1 4128:1 5427:1 5807:1 5990:1 5992:1 5994:1 6613:1 7277:1 7677:1
1 102:1 734:1 2160:1 2167:1 2337:1 2799:1 3295:1 3504:1 3750:1 5590:1 5642:1 6249:1 6510:1 6750:1 6800:1 7441:1
1 55:1 57:1 119:1 137:1 179:1 1013:1 2853:1 3065:1 4439:1 4784:1 4899:1 5593:1 5605:1 7196:1 8134:1 8147:1 8609:
1 268:1 859:1 936:1 938:1 940:1 2163:1 2583:1 2749:1 3821:1 4149:1 4835:1 5159:1 5161:1 5163:1 5165:1 8600:1 907:
1 958:1 1555:1 2204:1 2241:1 2459:1 2835:1 3090:1 3093:1 3361:1 3363:1 3904:1 3943:1 4226:1 4716:1 6267:1 6461:1
1 1269:1 1485:1 1922:1 3304:1 4445:1 5055:1 5057:1 5159:1 5161:1 5163:1 5165:1 5294:1 5303:1 5545:1 5588:1 5666:
1 55:1 57:1 137:1 179:1 1013:1 1276:1 2763:1 3023:1 3904:1 4699:1 4808:1 5268:1 5270:1 5400:1 5635:1 5648:1 5866
1 4797:1 5385:1 5816:1 5891:1 7454:1 7962:1 9501:1
1 959:1 1481:1 2579:1 4012:1 4226:1 4481:1 5175:1 5183:2 5626:4 5663:1 6087:1 6487:1 6496:1 6970:1 7421:1 7531:1
1 318:1 320:1 829:1 1052:1 1091:1 1093:1 1095:1 1097:1 2953:1 3150:1 5193:1 6215:1 6514:1 6516:1 7992:1 8168:1 8
1 88:1 120:1 143:1 286:1 1078:1 1984:1 2139:1 2373:1 2459:1 2815:1 4083:1 5749:1 7390:1 7916:1 8289:1 9051:1 950
1 50:1 52:1 88:1 101:1 120:1 143:1 209:1 374:1 547:1 617:1 936:1 938:1 940:1 947:1 974:1 1025:1 1247:1 1278:1 13
1 1241:1 2472:1 3023:1 3994:1 4228:1 4448:1 4450:1 4702:1 5092:1 5261:1 5284:1 5286:1 5400:1 5613:1 5615:1 5621:
1 1081:1 1083:1 1085:1 1087:1 1729:1 1914:1 2295:1 3120:1 3275:1 3412:1 4130:1 5337:1 5362:1 6321:1 6349:1 6733:
1 322:1 834:1 1618:1 2045:1 2068:1 2378:1 2533:1 3113:1 3115:1 4702:1 6309:1 6712:1 6714:1 6716:1 6718:1 6720:1
1 34:1 134:1 199:1 327:1 329:1 599:1 696:1 1081:1 1083:1 1085:1 1087:1 1159:1 2070:1 2072:1 2478:1 2853:1 3061:1
1 657:1 1158:1 1467:1 2272:1 2439:1 2747:1 4336:1 4563:1 4592:1 4716:1 4928:1 4990:1 5263:1 5539:1 6401:1 7512:1

```

Figura 4.8 Formato del archivo que almacena la representación de cada artículo

4.3.4 Uso de las herramientas de LIBSVM

Con la representación del texto hecha en vectores se desea entrenar el clasificador con los datos de entrenamiento. A continuación se mostrará el proceso de entrenamiento usando la herramienta LIBSVM para el set de datos de la tercera, el mismo procedimiento se realiza luego para el set de datos de Reuters.

4.3.4.1 Escalado de los datos

Antes de hacer el entrenamiento los investigadores proponen escalar linealmente los vectores para evitar dificultades numéricas en los cálculos [14], ya que los cálculos en los kernel dependen de productos entre los vectores de característica, valores numéricos muy altos en los atributos podrían provocar problemas. Por esto se recomienda escalar cada atributo en un rango de $[-1,1]$ o $[0,1]$. Pero considerando el caso de estudio en particular en donde se tienen muchas características (palabras) existen muchos valores cero en los vectores, estos valores como se explicó no se representan en los archivos, pero si se usara el rango $[-1,1]$ para escalar si quedarían representados, lo que solo significa demora el proceso de entrenamiento y predicción.

Por supuesto también se deben escalar los datos que se usaran para prueba (test), usando el mismo método de escalado aplicado a los datos de entrenamiento.

```

C:\libsvm>svm-scale -s range -l 0 -u 1 training.txt > train.scale
C:\libsvm>svm-scale -r range test.txt > test.scale

```

Figura 4.9 Escalamiento de datos

La ilustración muestra el proceso para escalar los datos de entrenamiento definiendo 0 como límite inferior y 1 como superior, además se salva un archivo con los parámetros de escalado para luego aplicar el mismo método con los datos de prueba.

```

1 27:1 29:1 269:1 2050:1 2834:1 4525:1 5635:1 9501:1
1 55:1 57:1 134:1 137:1 179:1 195:1 199:1 544:1 546:1 548
1 298:1 423:1 455:1 461:1 465:1 502:1 509:2 511:2 513:2 5
1 58:2 173:1 295:1 316:1 440:1 496:1 507:1 509:1 511:1 51
1 905:1 1563:1 1654:1 1680:1 2242:1 3373:1 3494:1 3923:1
1 1528:1 1822:1 2487:1 2744:1 2747:1 3207:1 4012:1 4084:1
1 291:1 759:1 1862:2 1864:2 3120:1 3225:2 3817:1 4135:1 4
1 418:1 1679:1 1757:1 1922:1 3030:1 3679:1 3681:1 3683:1
1 273:1 1239:1 1347:1 1875:1 2190:1 2192:1 2428:1 2794:1
1 401:1 1917:1 3120:1 3751:1 3753:1 3895:1 4111:1 4128:1
1 102:1 734:1 2160:1 2167:1 2337:1 2799:1 3295:1 3504:1 3
1 55:1 57:1 119:1 137:1 179:1 1013:1 2853:1 3065:1 4439:1
1 268:1 859:1 936:1 938:1 940:1 2163:1 2583:1 2749:1 3821
1 958:1 1555:1 2204:1 2241:1 2459:1 2835:1 3090:1 3093:1
1 1269:1 1485:1 1922:1 3304:1 4445:1 5055:1 5057:1 5159:1

```

Figura 4.3 Datos de entrenamiento originales

```

1 27:0.5 29:0.5 269:0.5 2050:1 2834:1 4525:1 5635:0.25
1 55:0.0714286 57:0.0714286 134:1 137:0.0714286 179:0.0
1 298:0.5 423:1 455:1 461:1 465:1 502:1 509:1 511:1 513
1 58:1 173:1 295:1 316:1 440:1 496:1 507:1 509:0.5 511:
1 905:1 1563:1 1654:1 1680:1 2242:1 3373:1 3494:1 3923:
1 1528:1 1822:1 2487:1 2744:0.5 2747:1 3207:1 4012:0.5
1 291:1 759:1 1862:0.5 1864:0.5 3120:1 3225:1 3817:0.33
1 418:0.5 1679:1 1757:1 1922:0.5 3030:0.5 3679:0.166667
1 273:1 1239:1 1347:1 1875:1 2190:0.5 2192:0.5 2428:1 2
1 401:1 1917:1 3120:1 3751:0.5 3753:0.5 3895:0.5 4111:1
1 102:1 734:1 2160:1 2167:1 2337:1 2799:1 3295:1 3504:1
1 55:0.0714286 57:0.0714286 119:1 137:0.0714286 179:0.0
1 268:1 859:1 936:1 938:1 940:1 2163:1 2583:1 2749:0.5
1 958:1 1555:1 2204:1 2241:1 2459:1 2835:1 3090:1 3093:
1 1269:1 1485:1 1922:0.5 3304:0.5 4445:1 5055:1 5057:1

```

Figura 4.4 Datos de entrenamiento escalados

4.3.4.2 Selección del kernel y distintos parámetros

Una vez escalados los datos de entrenamiento, se procede a entrenar el clasificador SVM eligiendo el kernel a usar y los distintos parámetros de estos.

La guía de LIBSVM sugiere considerar el kernel RBF (radial basis function) como primera opción y probar con el valor de su parámetro gamma (γ) junto con el parámetro C , que castiga la mala clasificación de los puntos del vector de entrenamiento, del SVM.

El procedimiento propuesto por la guía de clasificación con SVM [14] es:

1. Efectuar un escalado lineal de los datos de entrenamiento y prueba.
2. Considerar el Kernel RFB.
3. Usar validación cruzada para encontrar los mejores parámetros C y gama (γ).
4. Utilizar los mejores parámetros C y gamma encontrados para entrenar el clasificador.

5. Efectuar la prueba.

4.3.4.3 Validación cruzada (cross-validation)

Existen dos parámetros al usar el kernel RBF C y γ . En un principio no se sabe cuales valores deben tomar para obtener buenos resultados. Un método para encontrar para encontrar los parámetros más adecuados para obtener la mejor precisión es la validación cruzada, que hace uso de parte los datos de entrenamiento como datos de prueba, esto para poder obtener una clara medida accuracy ya que están previamente clasificados. En este método se define el parámetro v , v -fold cross-validation divide el set de entrenamiento en v subsets de igual tamaño. Secuencialmente se prueba un subset usando el clasificador entrenado con los otros $v-1$ sets.

La cross-validation puede prevenir el problema de overfitting, que es cuando no se ha aprendido muy bien la característica de los datos de entrenamiento, por lo que se hace una mala clasificación. La ilustración 30 muestra el problema.

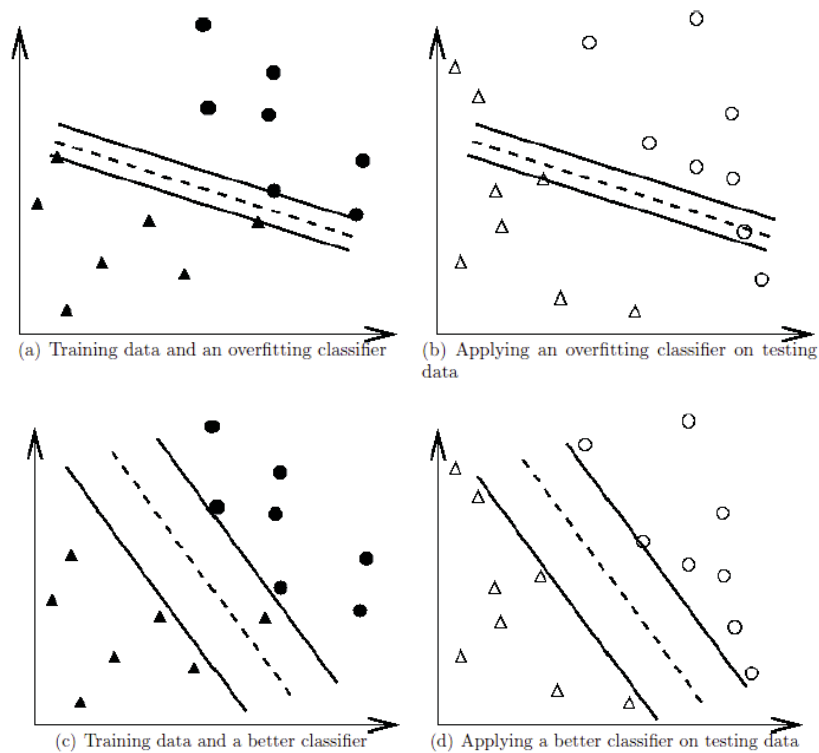


Figura 4.5. Comparación de clasificadores con y sin overfitting

LIBSVM proporciona un script en Python que hace la validación cruzada comparando distintos resultados para encontrar automáticamente los parámetros C y γ que mejor resultados puedan obtener en la clasificación.

Tras efectuar este proceso para los datos de entrada del caso de estudio se llegó a encontrar que los parámetros óptimos para este caso son con $C = 2048$ y $\gamma = 0,0001220703125$.

```
[local] 15 -3 73.1 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] -5 -3 58.9 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] 7 -3 73.1 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] 1 -3 81.4 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] 13 -7 82.8 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] 13 -1 53.1 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] 13 -13 87.8 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] 13 1 49.8 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] 13 -11 87.6 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] 13 -5 83.2 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] 13 -15 88.1 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] 13 3 50.0 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] 13 -9 83.0 (best c=2048.0, g=0.0001220703125, rate=88.2)
[local] 13 -3 73.1 (best c=2048.0, g=0.0001220703125, rate=88.2)
2048.0 0.0001220703125 88.2
tusk@tusk-laptop:~/libsvm-2.9/windows$ ./grid.py training.scale
```

Figura 4.6 Cross-validation

Cabe mencionar que si el número de características es de gran tamaño, como lo es con este caso de estudio, podría no ser necesario mapear los datos a una dimensionalidad mayor. Ya que el mapeo no lineal no mejoraría el desempeño. Por lo que usar el kernel lineal sería suficientemente bueno, y solo se busca el parámetro C . Se indica que el kernel RBF es tan bueno como el lineal, pero esto solo después de buscar un correcto C y γ . [14]

Por esto, además del kernel RBF con los parámetros encontrados tras la cross-validation, a modo de prueba se usarán distintos parámetros para C y γ con este RBF así como los otros kernel implementados en la herramienta LIBSVM para comparar sus resultados.

4.3.4.4 Entrenamiento del clasificador y pruebas

Para entrenar el clasificador se usa el archivo con los vectores de los datos de entrenamiento ya escalados, indicando las opciones de tipo de kernel y parámetros.

```
C:\libsvm>svm-train
Usage: svm-train [options] training_set_file [model_file]
options:
-s svm_type : set type of SVM (default 0)
  0 -- C-SVC
  1 -- nu-SVC
  2 -- one-class SVM
  3 -- epsilon-SUR
  4 -- nu-SUR
-t kernel_type : set type of kernel function (default 2)
  0 -- linear: u'*v
  1 -- polynomial: (gamma*u'*v + coef0)^degree
  2 -- radial basis function: exp(-gamma*|u-v|^2)
  3 -- sigmoid: tanh(gamma*u'*v + coef0)
  4 -- precomputed kernel (kernel values in training_set_file)
-d degree : set degree in kernel function (default 3)
-g gamma : set gamma in kernel function (default 1/num_features)
-r coef0 : set coef0 in kernel function (default 0)
-c cost : set the parameter C of C-SVC, epsilon-SUR, and nu-SUR (default 1)
-n nu : set the parameter nu of nu-SVC, one-class SVM, and nu-SUR (default 0.5)
-p epsilon : set the epsilon in loss function of epsilon-SUR (default 0.1)
-m cachesize : set cache memory size in MB (default 100)
-e epsilon : set tolerance of termination criterion (default 0.001)
-h shrinking : whether to use the shrinking heuristics, 0 or 1 (default 1)
-b probability_estimates : whether to train a SVC or SUR model for probability estimates, 0 or 1 (default 0)
-wi weight : set the parameter C of class i to weight*C, for C-SVC (default 1)
-v n : n-fold cross validation mode
-q : quiet mode (no outputs)
C:\libsvm>
```

Figura 4.7 Opciones de LIBSVM

```

C:\WINDOWS\system32\cmd.exe
C:\libsvm>svm-scale -s range -l 0 -u 1 training.txt > train.scale
C:\libsvm>svm-scale -r range test.txt > test.scale
C:\libsvm>svm-train -t 2 -c 2048 -g 0.0001220703125 train.scale
.*
optimization finished, #iter = 1885
nu = 0.102874
obj = -107246.677591, rho = 6.880334
nSU = 792, nBSU = 3
Total nSU = 792
C:\libsvm>svm-predict test.scale train.scale.model test.predict
Accuracy = 85.6% (428/500) (classification)
C:\libsvm>

```

Figura 4.8 Proceso completo de clasificación

Proceso completo escalado de los datos de entrenamiento, escalado de los datos de prueba, entrenamiento del clasificador con el kernel RBF con $C=2048$ gamma = 0.0001220703125 y predicción con los datos de prueba obteniendo un 85.6% de accuracy.

Para hacer una comparación de accuracy entre las distintas representaciones, los distintos kernel, además de los parámetros de cada kernel se realizó un script en python que, usando el software Libsvm, realizó el entrenamiento y las pruebas en reiteradas ocasiones cambiando cada vez entre los distintos kernel y los parámetros de estos kernel con los que se entrenaba la máquina de aprendizaje, a continuación se presentan los resultados finales en un gráfico exponiendo los mejores accuracy del set de datos de La Tercera para cada representación.

Tabla 4.1 Tabla Resultados Accuracy SVM Set La Tercera

	Lineal	Polinomial	RBF	Sigmoidal
Frecuencia	86%	78,60%	86,20%	87,80%
TF	90%	69,20%	87,60%	90,20%
TF*IDF	88,60%	71,40%	89%	89,20%

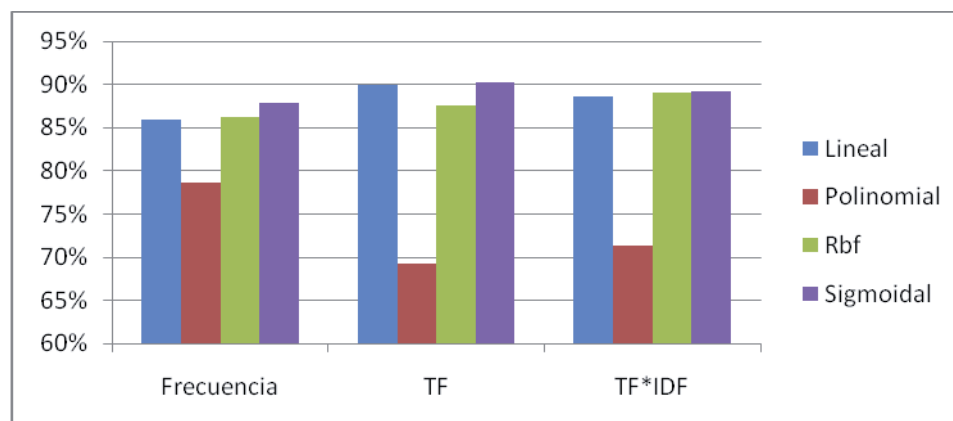


Figura 4.9 Resultados Accuracy SVM Set La Tercera

Además de las pruebas con el clasificador SVM se utilizó el clasificador Bayes Ingenuo con el mismo set de datos pero utilizando en este caso solamente la representación de frecuencia.

Tabla 4.2 Resultados Accuracy Bayes Set La Tercera

	Bayes
Frecuencia	84,80%

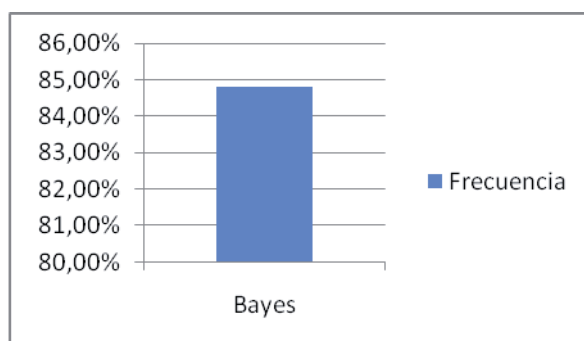


Figura 4.10 Resultados Accuracy Bayes La Tercera

4.4 Pruebas sobre colección Reuters-21578

Luego de realizar las pruebas con el set de datos extraídos del periódico en línea “La Tercera” se realizó un estudio acabado con una parte de los datos de la colección de Reuters 21578 con el fin de validar el proceso realizado anteriormente y comparar los resultados con los obtenidos en nuestras pruebas, esta colección contiene documentos que contienen noticias de la agencia en formato XML, contiene cinco clases “abstractas”: Topics, Places, People, Exchanges y Orgs. Estos documentos son multi-label es decir cada documento está etiquetado como perteneciente a una o más categorías. Es por esto que se hizo una adaptación a single label tomando documentos dentro de la clase abstracta “topics” que estuvieran clasificados en las categorías acq, que representa noticias de adquisiciones de empresas, y de la categoría earn, referente a ganancias de empresas.

Se hicieron dos clasificaciones con este set de datos, clasificar los documentos en las categorías acq vs no acq y clasificar en acq vs earn. Para esto se probó con tres representaciones de texto; Frecuencia, Frecuencia normalizada (Term Frequency, TF) y Term Frequency – Inverse Document Frequency (TF-IDF), además de los cuatro kernel más usados; Kernel Lineal, Kernel Polinomial, Kernel Radial Basis Function y Kernel Sigmoidal. También se consideraron los parámetros de cada kernel, variando solo el valor de C en el caso del kernel lineal y los valores de C y Γ para los demás kernel.

A continuación se presentarán los resultados del estudio presentando el **Accuracy** alcanzado en cada caso luego de entrenar con el clasificador SVM con los parámetros antes

mencionados (Tipo de representación y tipo de kernel). El proceso de entrenamiento y predicción para obtener los accuracy es el mismo utilizado en el caso de estudio anterior.

Tabla 4.3 Resultados Svm Set Reuters

Representación	Set	Accuracy	Kernel
Frecuencia	Acq vs no Acq	95,20%	Sigmoidal
Frecuencia	Acq vs earn	96,20%	Sigmoidal
TF	Acq vs no Acq	96,00%	RBF
TF	Acq vs earn	97,00%	Sigmoidal
TF*IDF	Acq vs no Acq	94,20%	Sigmoidal
TF*IDF	Acq vs earn	95,60%	Sigmoidal

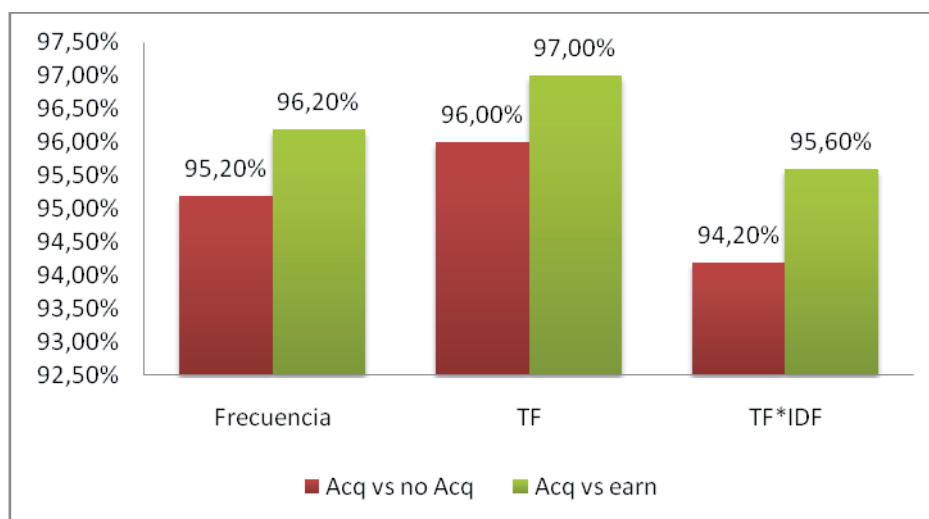


Figura 4.11 Resultados Accuracy Svm Set Reuters

4.5 Conclusiones de las pruebas

Tras realizar las pruebas con los datos de estudio, en forma general, se puede concluir que el número de datos de entrenamiento fue suficiente para encontrar resultados aceptables de accuracy (alrededor de 89%). Estos resultados se obtuvieron con un parámetro C alrededor de 0,5 lo que se considera bueno ya que los datos no están penalizados con exageración lo que no sería bueno si se quieren clasificar nuevos documentos muy diferentes a los que se usaron para entrenar.

Se pueden sacar conclusiones individuales para cada set de datos, para las representaciones y para cada uno de los kernel probados. La tabla siguiente presente los mejores resultados obtenidos por cada representación y set de datos.

Tabla 4.4 Resultados Accuracy Svm

Representación	Set	Accuracy	Kernel	C	Gamma
Frecuencia	Tercera	87,8%	Sigmoidal	0,5	0,7
Frecuencia	Acq vs no Acq	95,2%	Sigmoidal	0,1	0,7
Frecuencia	Acq vs earn	96.2%	Sigmoidal	100	0,001
TF	Tercera	90,2%	Sigmoidal	5	0,01
TF	Acq vs no Acq	96,0%	RBF	50	0,001
TF	Acq vs earn	97,0%	Sigmoidal	1	0,1
TF*IDF	Tercera	89,2%	Sigmoidal	1	0,05
TF*IDF	Acq vs no Acq	94.2%	Sigmoidal	0,1	0,1
TF*IDF	Acq vs earn	95.6%	Sigmoidal	0,1	0,7

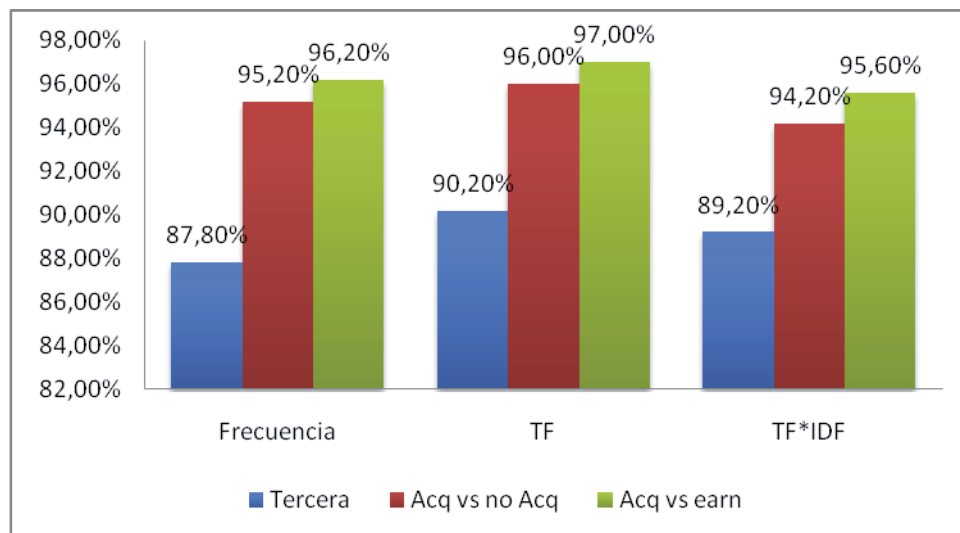


Figura 4.12 Resultados Accuracy Svm

Para el caso de los set de datos se puede apreciar que los datos de Reuters obtuvieron mejores resultados, esto se puede atribuir a que son datos más homogéneos y específicos en la forma en que están escritos, en cambio los de La tercera son más heterogéneos, además de aprecia que es más eficaz clasificar una clase versus otra que clasificar una clase versus su complemento.

Para el caso de la representación se aprecia que la de Term Frequency (TF) resulta la de mayores accuracy pero con un margen pequeño de diferencia, el resultado sorprende en primera instancia ya que se asume que la representación TF-IDF es mejor que la TF y esta a la vez mejor que Frecuencia.

En el caso del kernel la decisión parece ser más clara y según estas pruebas se podría afirmar que el mejor kernel para la clasificación de texto es el Sigmoidal. Tal vez como coincidencia se llegó a esta tendencia ya que también se obtuvieron valores de accuracy

cercanos a los máximos con los kernel lineal y Rbf. Lo único que se puede afirmar con certeza es que según estas pruebas el kernel Polinomial no resulta adecuado para la clasificación de texto.

4.5.1 Comparación Caso de estudio-Reuters 21578

Para poder evaluar las distintas representaciones usadas, se hace necesario comparar resultados con estudios sobre colecciones conocidas, es por ello que se trabajó también en Reuters 21578, la mejor forma de realizar esta comparación es realizando evaluaciones formales sobre la clasificación entregada por las máquinas de aprendizaje utilizadas, las medidas que se usarán son las que se detallaron en el capítulo 3.4.6, siendo los resultados finales los que se muestran a continuación:

Evaluación SVM

Se tomaron en cuenta los resultados con mejor rendimiento de accuracy para cada representación tanto para el caso de estudio de la tercera como para las pruebas sobre Reuters. Estos datos se pueden encontrar en la tabla anterior “Mejores resultados”.

Tabla 4.5 Evaluación SVM

Representación	DataSet	Accuracy	F ₁	Precision	Recall
Frecuencia	Acq vs No Acq	95,2%	96,8%	93,8%	96,8%
Frecuencia	Acq vs Earn	96,2%	97,0%	94,2%	98,4%
Frecuencia	La Tercera	87,8%	95,6%	91,6%	83,2%
TF	Acq vs No Acq	96,0%	98,7%	97,5%	94,4%
TF	Acq vs Earn	97,0%	98,0%	96,1%	98,0%
TF	La Tercera	90,2%	96,6%	93,5%	86,4%
TF*IDF	Acq vs No Acq	94,2%	97,0%	90,8%	98,4%
TF*IDF	Acq vs Earn	95,6%	95,8%	91,9%	100%
TF*IDF	La Tercera	89,2%	96,2%	92,6%	85,2%

Evaluación Bayes Ingenuo

En este caso en particular la representación ocupada fue solamente Frecuencia, los resultados se muestran en la tabla.

Tabla 4.6 Evaluación Bayes

Representación	DataSet	Accuracy	F ₁	Precision	Recall
Frecuencia	Acq vs No Acq	83,0%	96,8%	96,6%	69,7%
Frecuencia	Acq vs Earn	89,2%	96,9%	94,1%	85,3%
Frecuencia	La Tercera	84,8%	88,6%	79,6%	93,6%

5 Definición del sistema

5.1 Introducción

Una vez terminada la etapa de investigación y pruebas sobre los algoritmos escogidos para realizar la clasificación de texto, se hace necesario desarrollar un sistema que otorgue la posibilidad de llevar a la práctica la aplicación de una máquina de aprendizaje sobre el dominio estudiado.

La finalidad de este sistema es ser una herramienta de apoyo académico para usuarios que estén insertos en el tema de la clasificación de texto usando máquinas de aprendizaje, dándole la posibilidad de entrenar una máquina de aprendizaje con distintos parámetros y así obtener los mejores resultados en la clasificación.

Antes de especificar el sistema en sí, al igual que cualquier software debe existir una manera de abordarlo, para ello es trascendental escoger que modelo de proceso y que metodología de desarrollo utilizar.

5.2 Elección metodología de desarrollo

La metodología de desarrollo escogida es la de Orientación a objetos cuyas principales características y el motivo de su elección se describirán a continuación

La Orientación a Objetos se define como un enfoque de desarrollo de software que se caracteriza por su capacidad para modelar o representar las entidades de un dominio de aplicación en una forma natural y directa, en términos de objeto, este objeto es una unidad de software que modela o representa el estado y la dinámica que tiene una entidad. Los objetos simulan la estructura del dominio de aplicación y también su comportamiento, esto es posible gracias al “paso de mensajes”, en donde los objetos se comunican y cooperan entre sí.

Tiene asociadas ventajas [14]

- Mejora la captura y validación de requerimientos
- Acerca el “espacio del problema” y el “espacio de la solución”
- Facilita la construcción, el mantenimiento y la reutilización
- Facilita la transición entre las distintas fases
- Disipa la barrera entre el qué y el cómo

La elección de esta metodología está muy asociada a sus ventajas y es que proporciona una mejor forma de validar requerimientos, el modelado se “acerca a la realidad”, facilitando la comprensión de la problemática y su solución, y pensando en la etapa de codificación, la modificación de objetos es más simple, y si éstos son correctamente diseñados pueden ser reutilizados.

Una vez escogido modelo de proceso y metodología de desarrollo, se puede partir con el desarrollo del software, iniciando con las etapas 1 y 2 definidas en 9.1, siendo la especificación de requerimientos la actividad seleccionada para resumirlas.

5.3 Especificación de requerimientos

5.3.1 Requerimientos funcionales

5.3.1.1 Requerimientos Usuario

Tareas relacionadas al entrenamiento:

- Podrá elegir que máquina de aprendizaje utilizar para el entrenamiento
- Al momento de entrenar podrá elegir que transformación se hará sobre los documentos dando como opción las siguientes alternativas de funciones de peso para modificar el vector de documentos: Binaria, Frecuencia, TF, TF*IDF, TF*RF.
- Antes del entrenamiento podrá elegir la opción de pre-procesamiento del documento, con la posibilidad de aplicar Remoción de Stopwords o no.
- Podrá determinar el número de elementos del conjunto de training y test.
- Podrá agregar categorías y los respectivos documentos asociados para entrenar la máquina bajo esa categoría.
- En el caso de elegir SVM, podrá elegir qué tipo de Kernel usar para el entrenamiento, y para cada Kernel elegir parámetros propios.
- Podrá crear distintos vectores de representación para los documentos.
- Podrá elegir con qué medida evaluar, accuracy, Precision, Recall, F_1

Relacionadas a la clasificación

- Podrá escoger bajo que categoría predeterminada por el sistema desea clasificar y agregar el conjunto de documentos necesario para realizar el proceso
- Obtendrá un resumen con los resultados de la clasificación.
- Podrá escoger bajo que categoría(s) predeterminadas por el sistema desea clasificar y agregar el conjunto de documentos necesario para realizar el proceso.
- Podrá escoger que máquina de aprendizaje se usará para la clasificación.

5.3.1.2 Caso de uso general del sistema

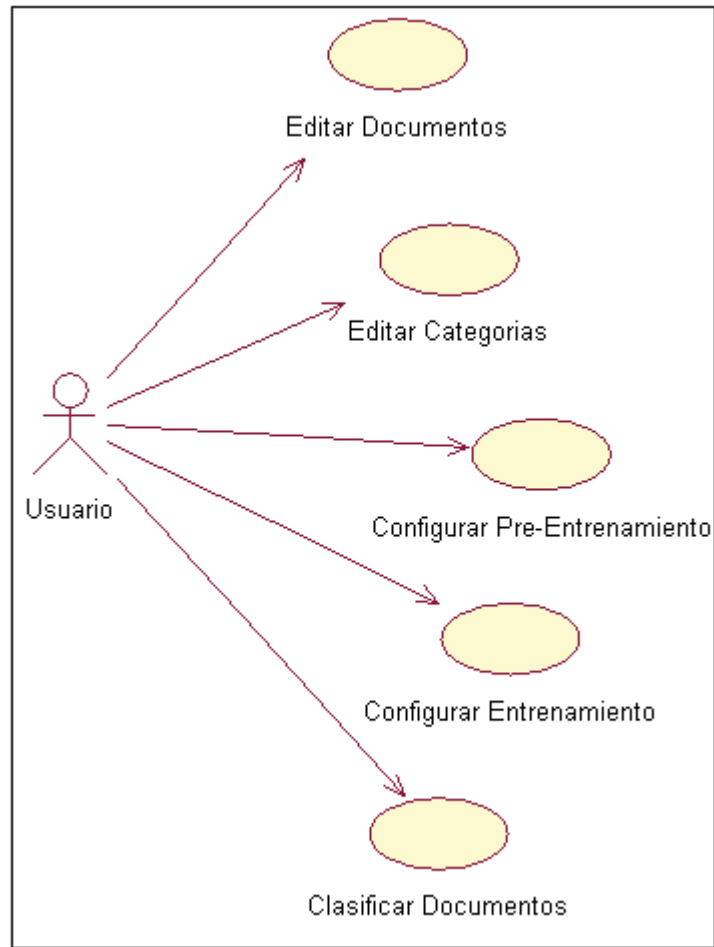


Figura 5.1 Caso de uso General del Sistema

5.3.1.3 Caso de uso “Configurar Pre- Entrenamiento”

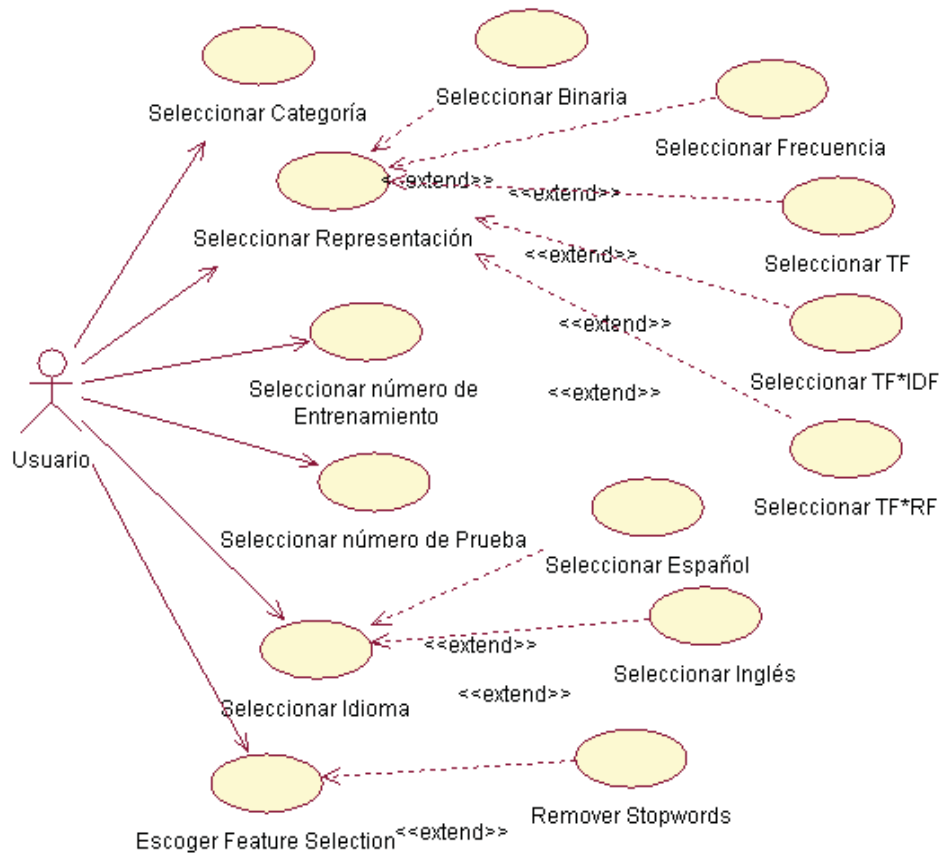


Figura 5.2 Caso de uso “Configurar Pre- Entrenamiento”

5.3.1.4 Caso de uso “Editar Categorías”

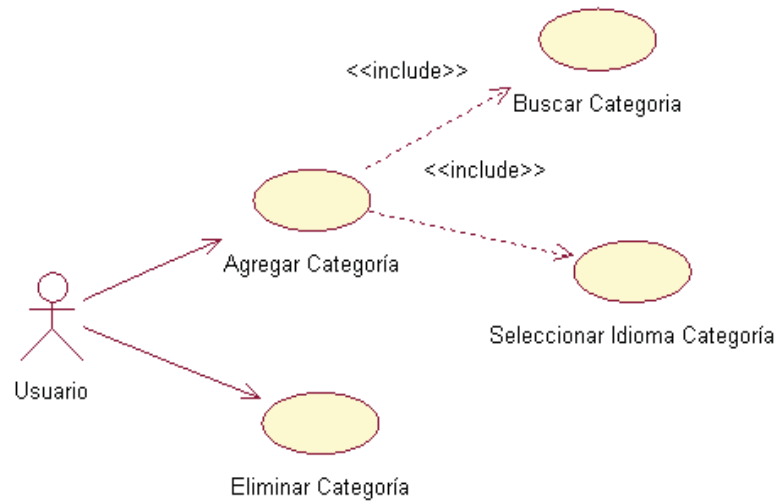


Figura 5.3 Caso de uso “Editar Categorías”

5.3.1.5 Caso de uso “Clasificar Documentos”

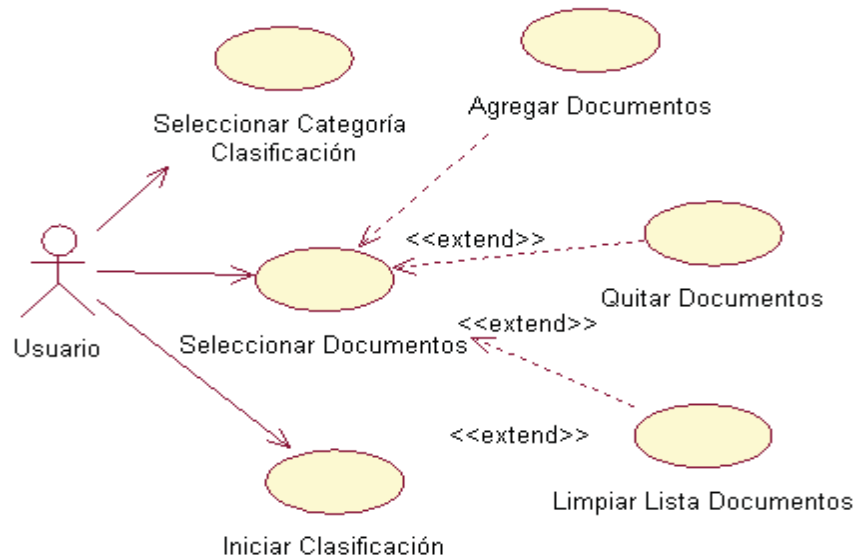


Figura 5.4 Caso de uso “Clasificar Documentos”

5.3.1.6 Caso de uso “Editar Documentos”

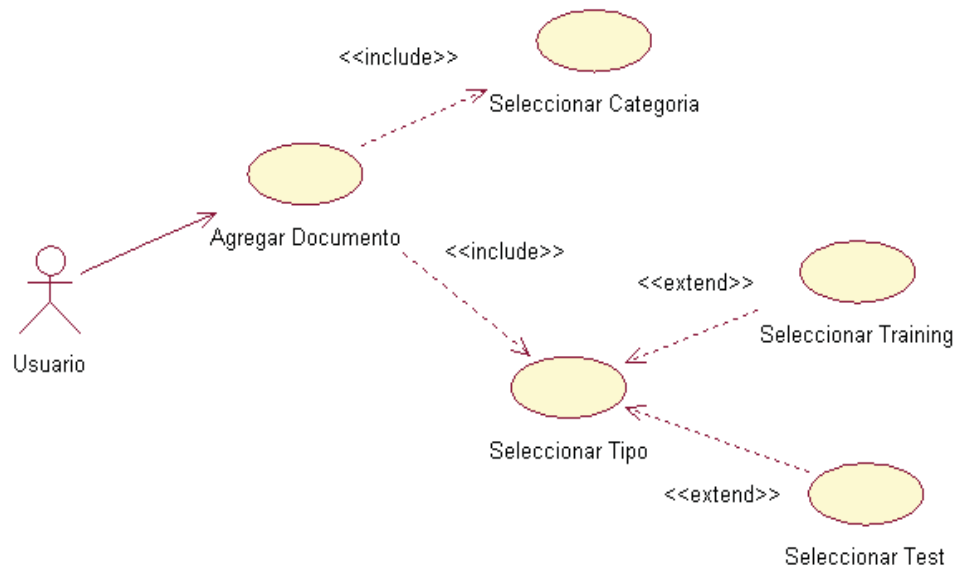


Figura 5.5 Caso de uso “Editar Documentos”

5.3.1.7 Caso de uso “Configurar Entrenamiento”

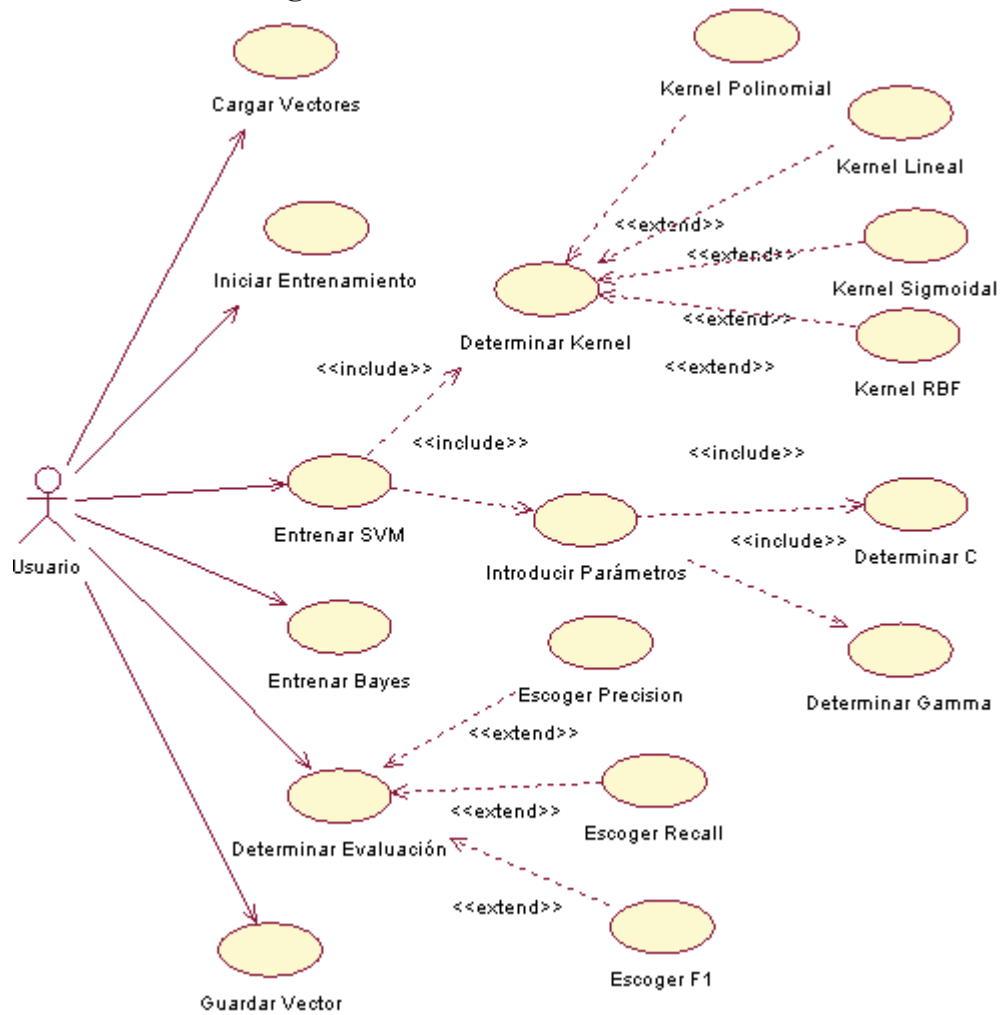


Figura 5.6 Caso de uso “Configurar Entrenamiento”

5.3.2 Requerimientos no funcionales

- El sistema deberá tener una interfaz gráfica para la carga de archivos de texto.
- El sistema deberá ser compatible con Gnu/Linux y Windows.
- Se deberá documentar el sistema en cada una de sus etapas de desarrollo.
- El sistema deberá desarrollarse con herramientas open source que no necesiten de licenciamiento.
- El sistema deberá ser portable a cualquier otro equipo que cumpla con los requerimientos mínimos de sistema.

5.4 Modelado del sistema

5.4.1 Diagrama de clases

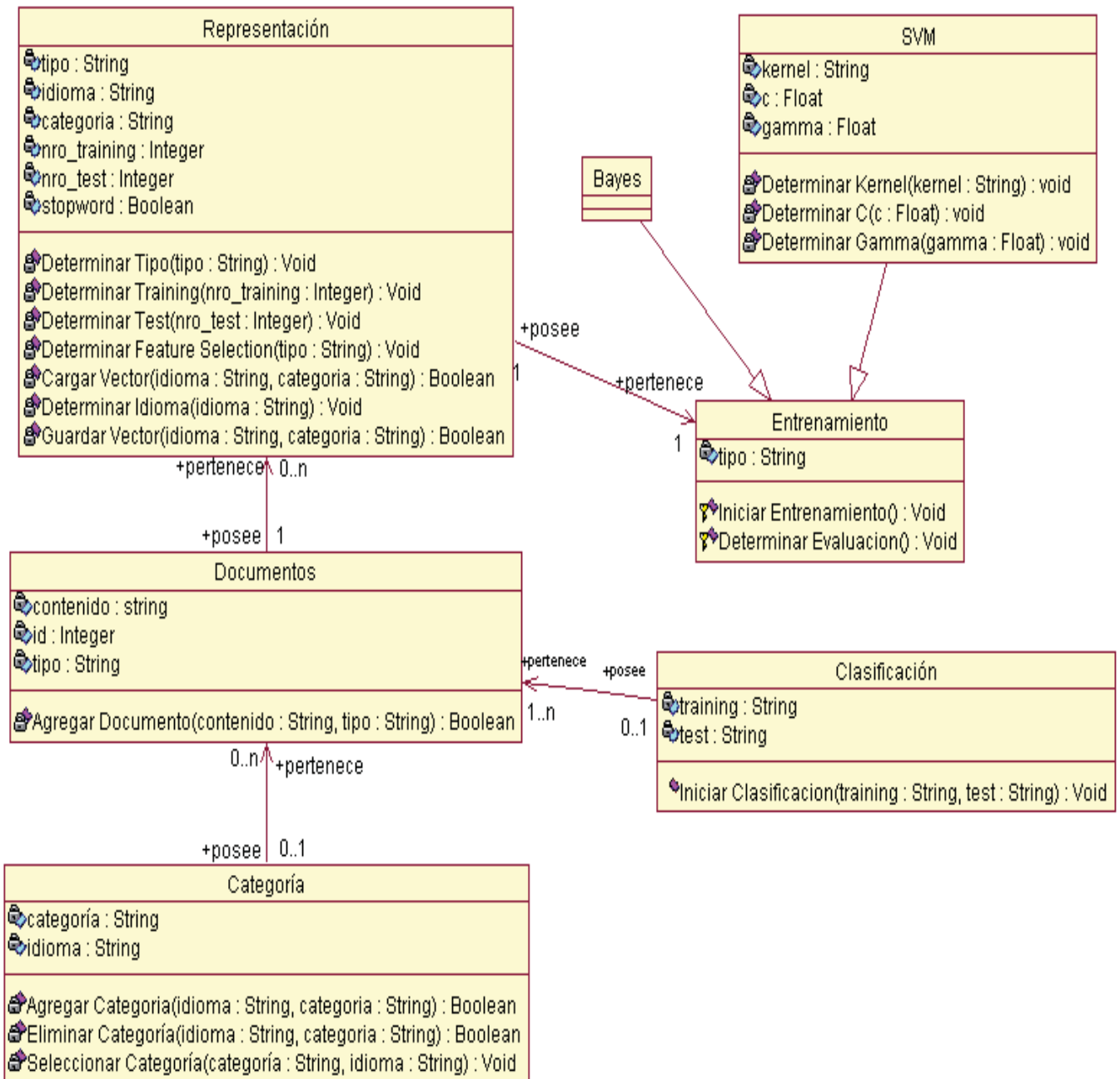


Figura 5.7 Diagrama de clases

5.4.2 Diagramas de secuencia

5.4.2.1 “Configurar Pre- Entrenamiento”

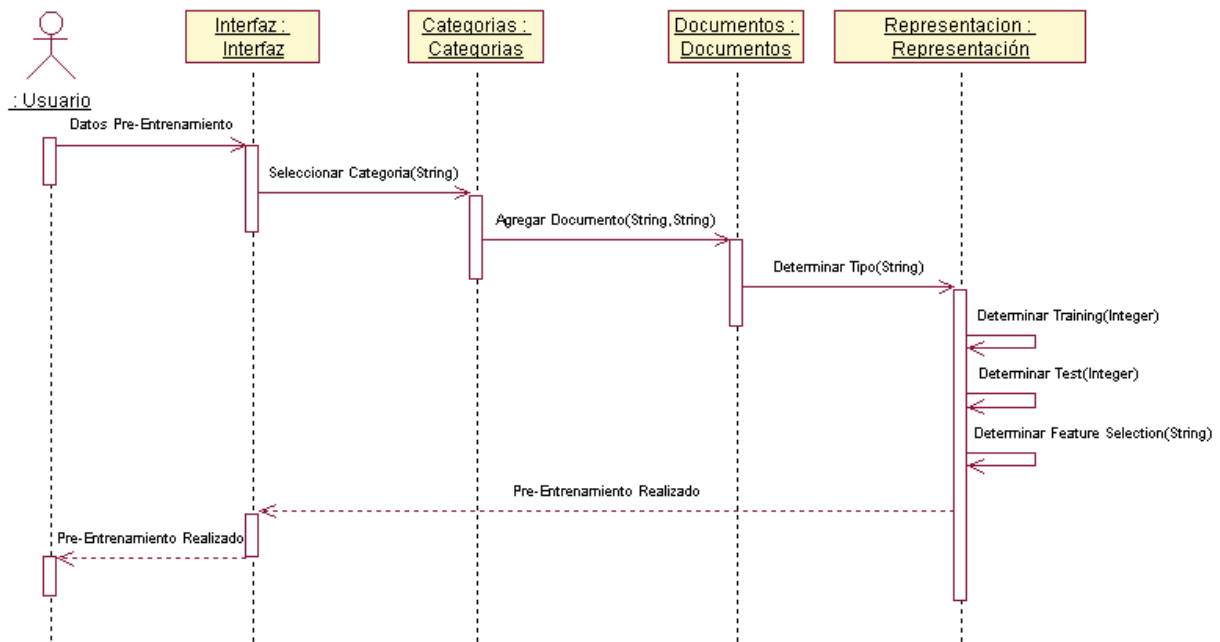


Figura 5.8 “Configurar Pre- Entrenamiento”

5.4.2.2 “Editar Categorías”

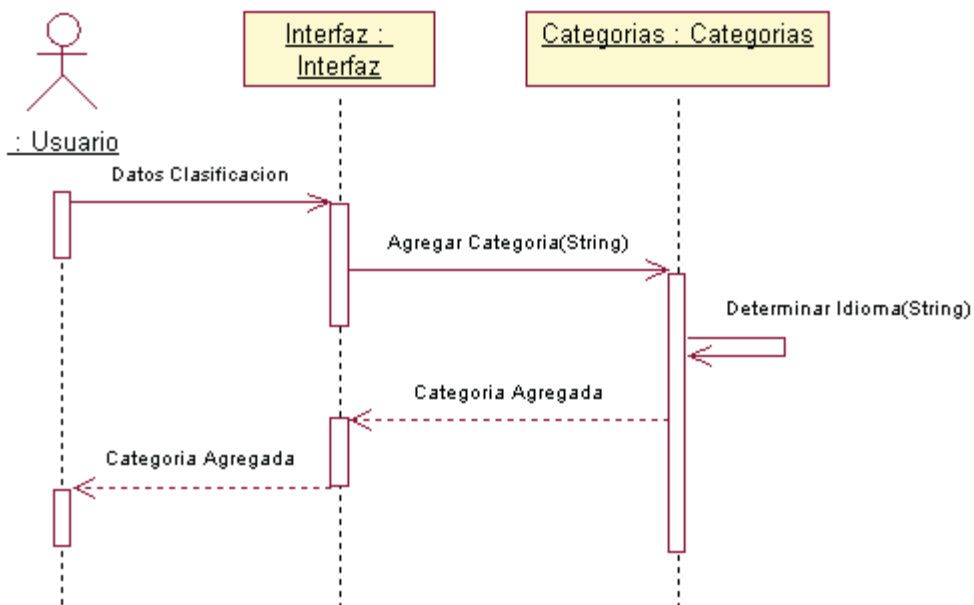


Figura 5.9 “Editar Categorías”

5.4.2.3 “Clasificar Documentos”

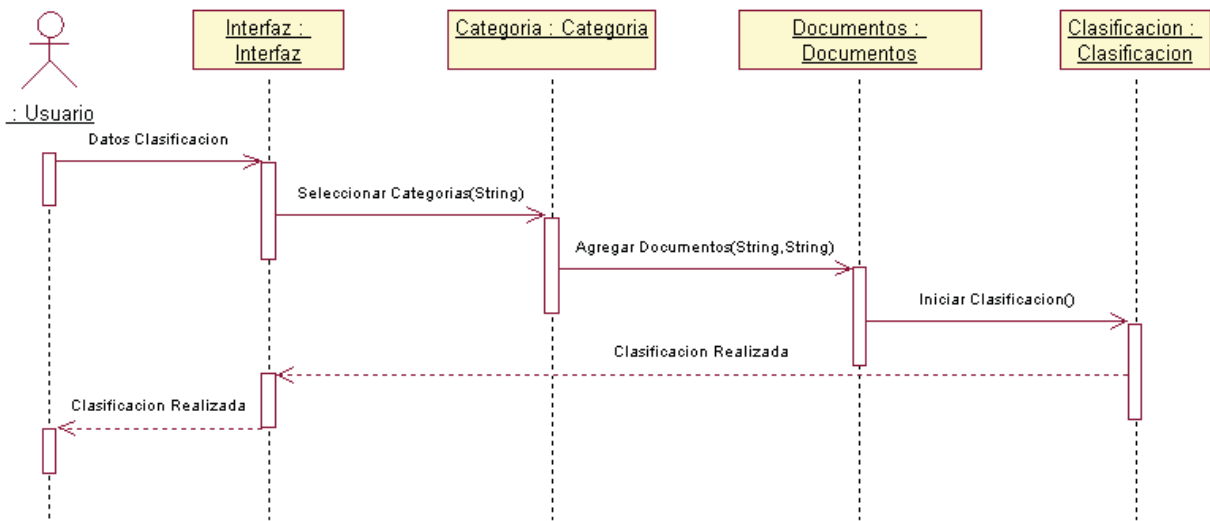


Figura 5.10 “Clasificar Documentos”

5.4.2.4 “Configurar Entrenamiento Bayes”

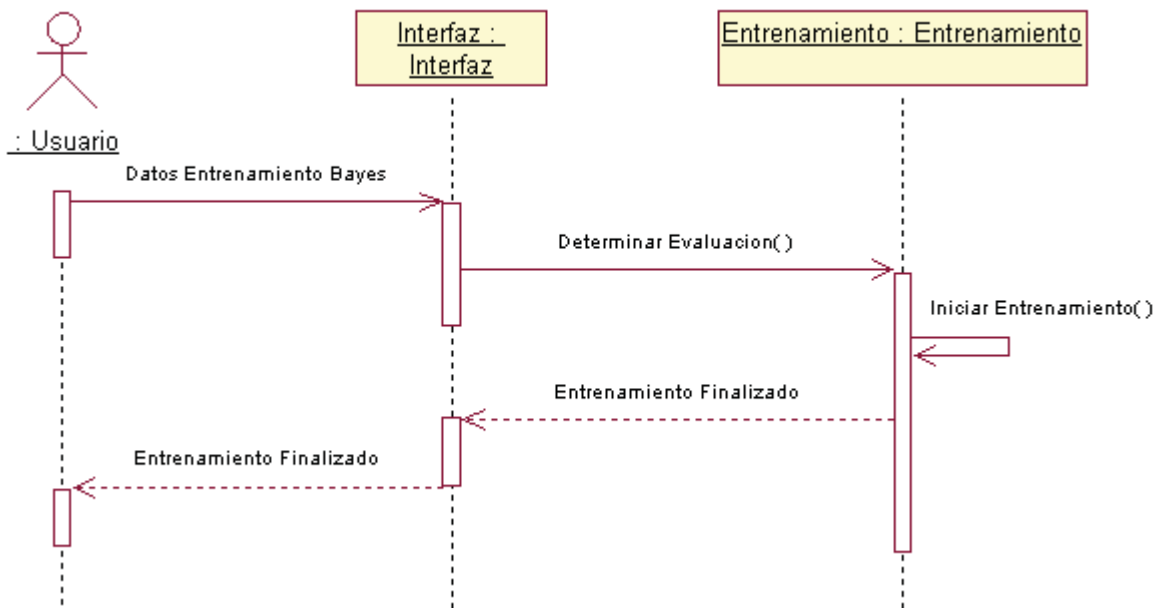


Figura 5.11 “Configurar Entrenamiento Bayes”

5.4.2.5 “Configurar Entrenamiento SVM”

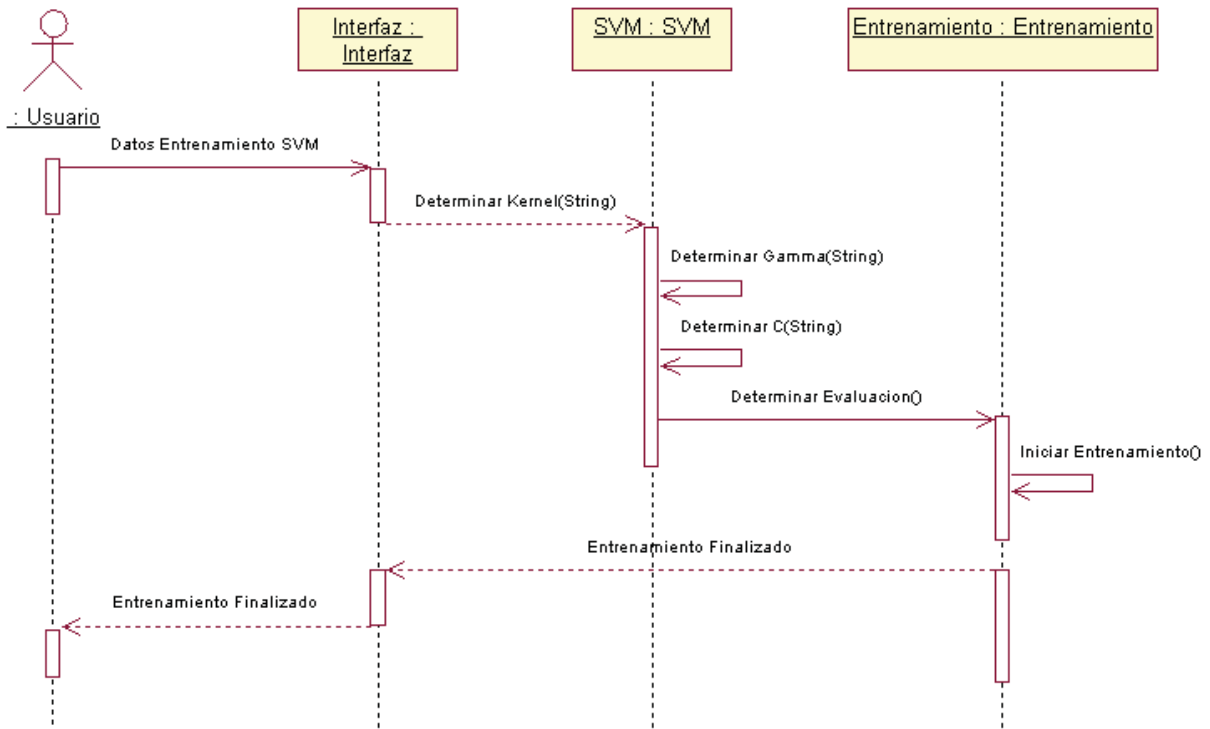


Figura 5.12 “Configurar Entrenamiento SVM”

5.5 Descripción del software clasificador

El software clasificador se construyó usando el framework para c++ de Nokia llamado Qt usando el entorno de desarrollo Qt Creator con el fin de apoyar la creación de una interfaz gráfica. A continuación se expondrán las pantallas del software explicando brevemente su utilización y luego se mostrará las secuencias de acciones posibles con el software a través de un diagrama de estado.

5.5.1 Interfaz Pre-Entrenamiento

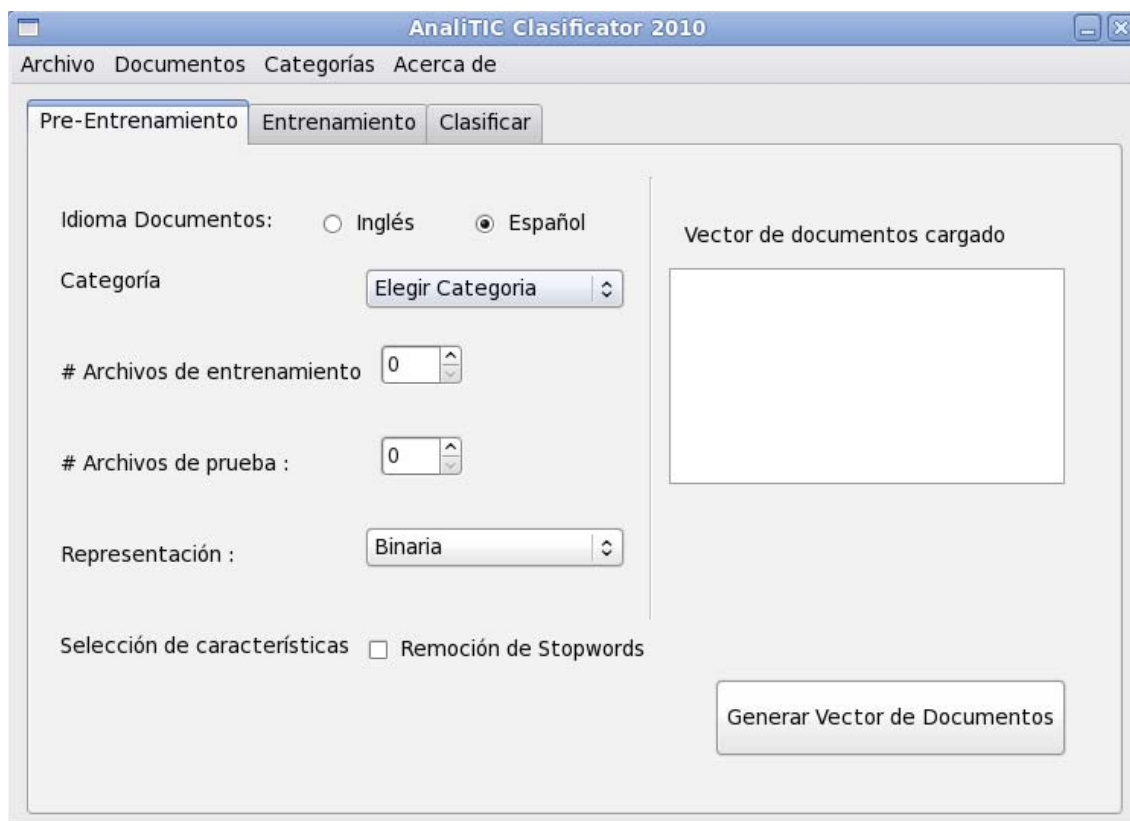


Figura 5.13 Interfaz Pre-Entrenamiento

En esta pantalla se genera la representación de los documentos de una categoría que puede ser elegida entre las que existen, diferenciando las categorías de idioma inglés y español. Para generar esta representación se deben elegir algunos parámetros, estos son; el número de archivos de entrenamiento y de prueba con el que se generará la representación; el tipo de representación que se generará con estos archivos, puede ser binaria, frecuencia, TF, TF*IDF o TF*RF.

Al generar la representación, esta se guarda en la base de datos y queda seleccionada como representación actual para poder ser usada en el entrenamiento.

5.5.2 Interfaz Entrenamiento

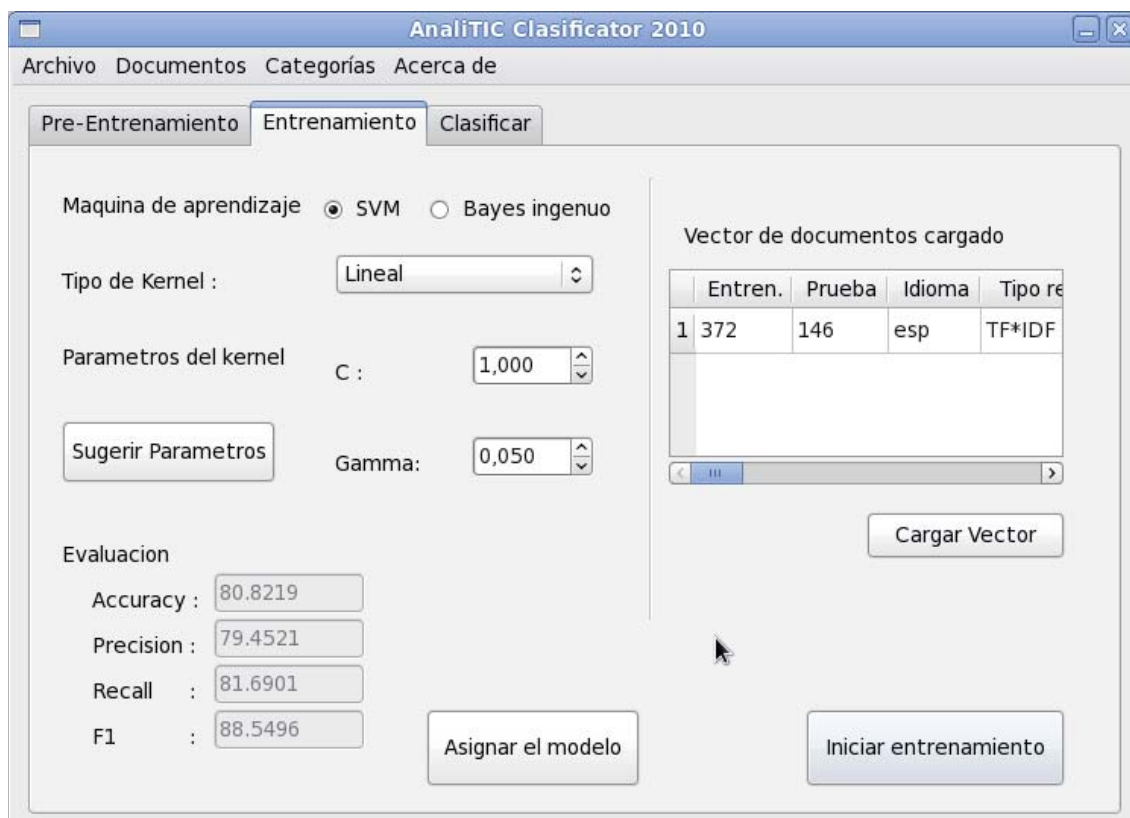


Figura 5.14 Interfaz Entrenamiento

En esta pantalla se realiza el entrenamiento utilizando la representación actual cargada diferenciando que máquina de aprendizaje usar SVM o Bayes Ingenuo. En caso de no haber hecho la representación anteriormente se puede cargar una representación salvada en la base de datos. Además se puede usar la opción de “Sugerir Parámetros” en la que se hará una cross-validation de los parámetros y dará como resultado los parámetros C y Gamma más convenientes para el entrenamiento según el kernel y el vector de documentos cargados.



Figura 5.15 Parámetros sugeridos por la cross validation

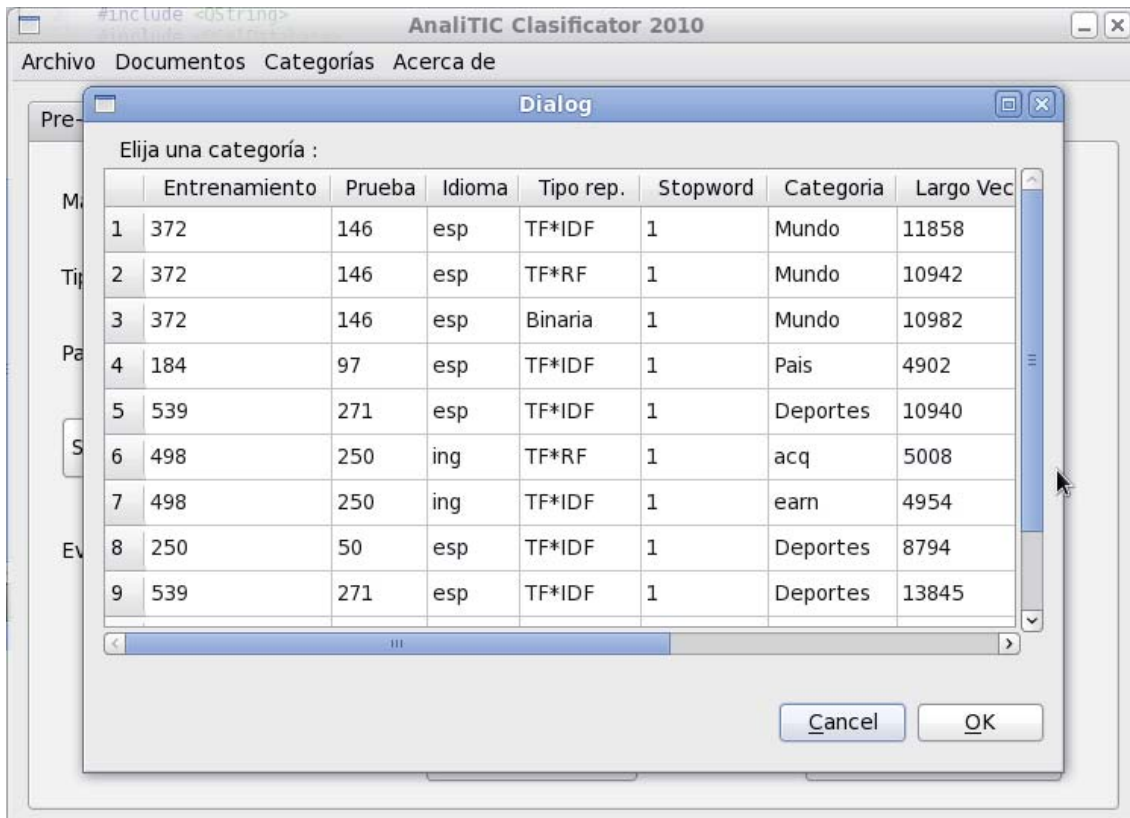


Figura 5.16 Interfaz Cargar Representación

Para iniciar el entrenamiento, además de tener una representación cargada (representación actual) se necesita elegir algunos parámetros. En el caso de SVM se elige el tipo de kernel que puede ser lineal, polinomial, RBF o Sigmoidal y el valor de los parámetros C y Gamma de estos kernel. Para Bayes tanto como para SVM.

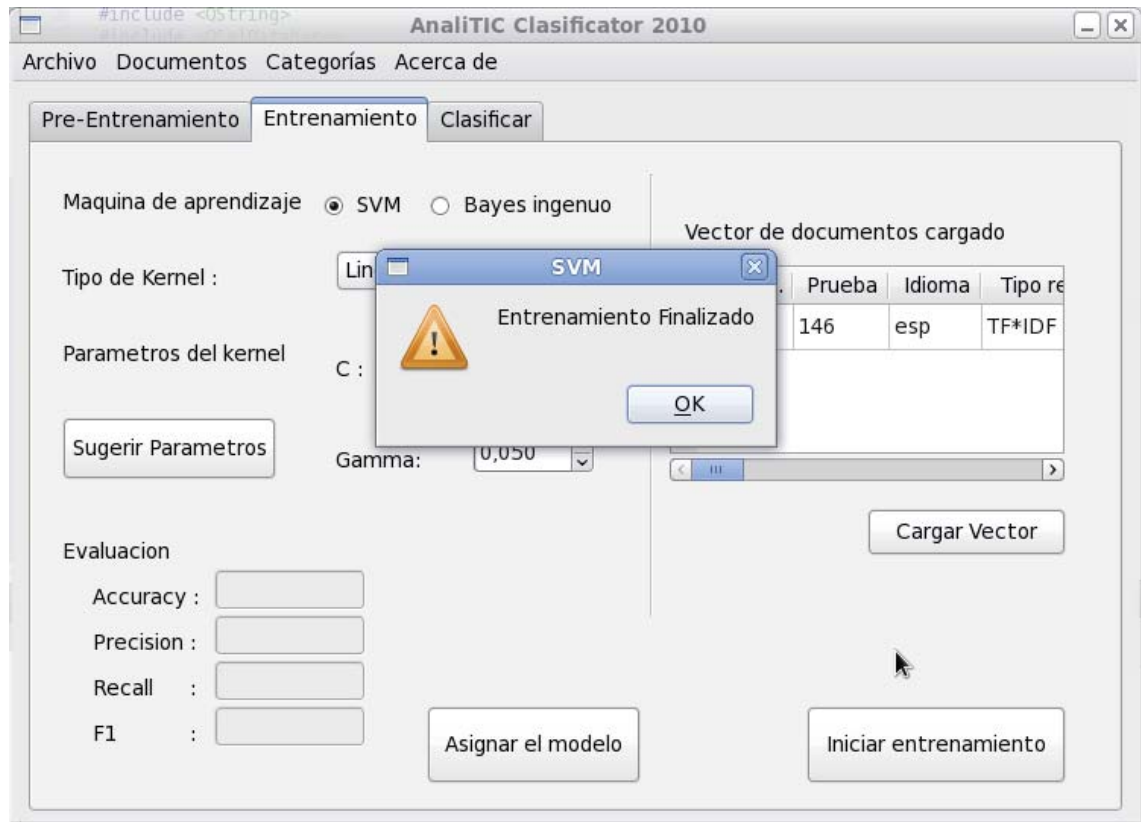


Figura 5.17 Entrenamiento Finalizado

Al finalizar el entrenamiento se mostrará la evaluación realizada con los documentos destinados para ello. Mostrando los valores de Accuracy, Precision, Recall y F₁.

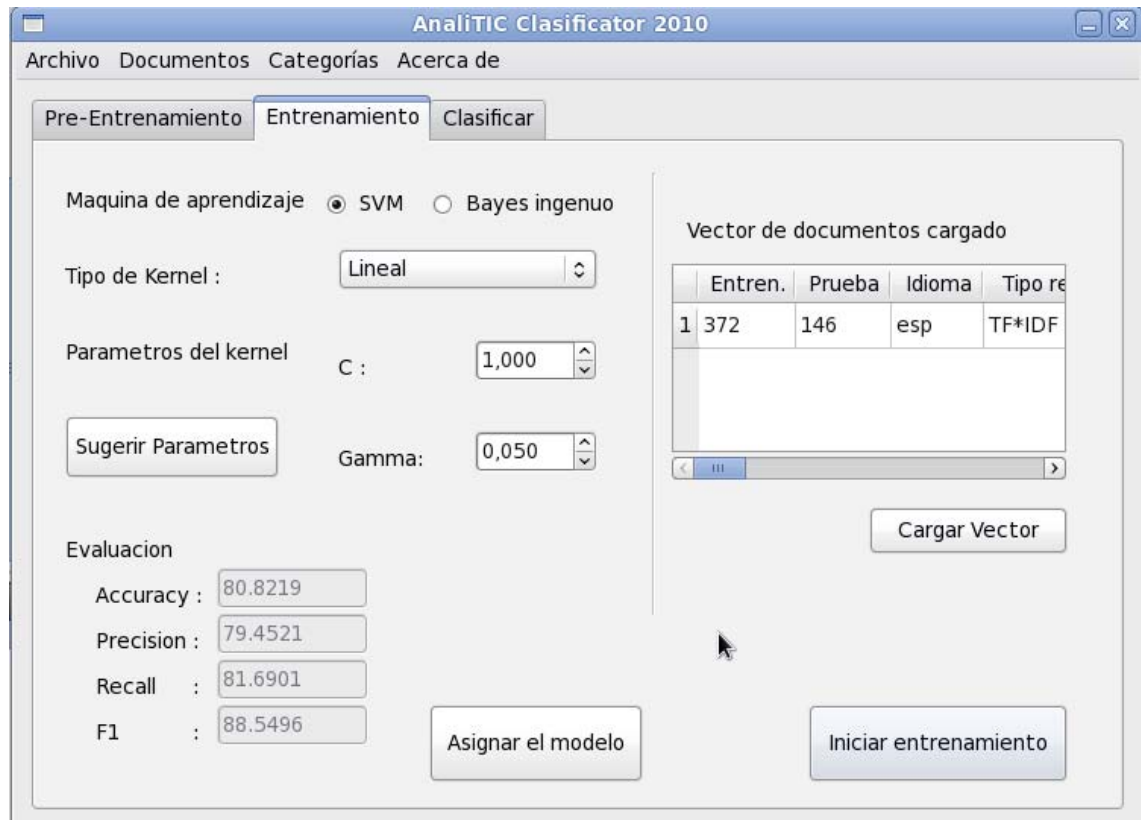


Figura 5.18 Resultados de la evaluación

Finalmente dependiendo de los resultados obtenidos en la evaluación del entrenamiento y con los conocimientos del operador del software se puede asignar el modelo generado por SVM o Bayes, con su Kernel y sus parámetros, a la categoría elegida del vector de documentos cargado.

5.5.3 Interfaz Clasificar

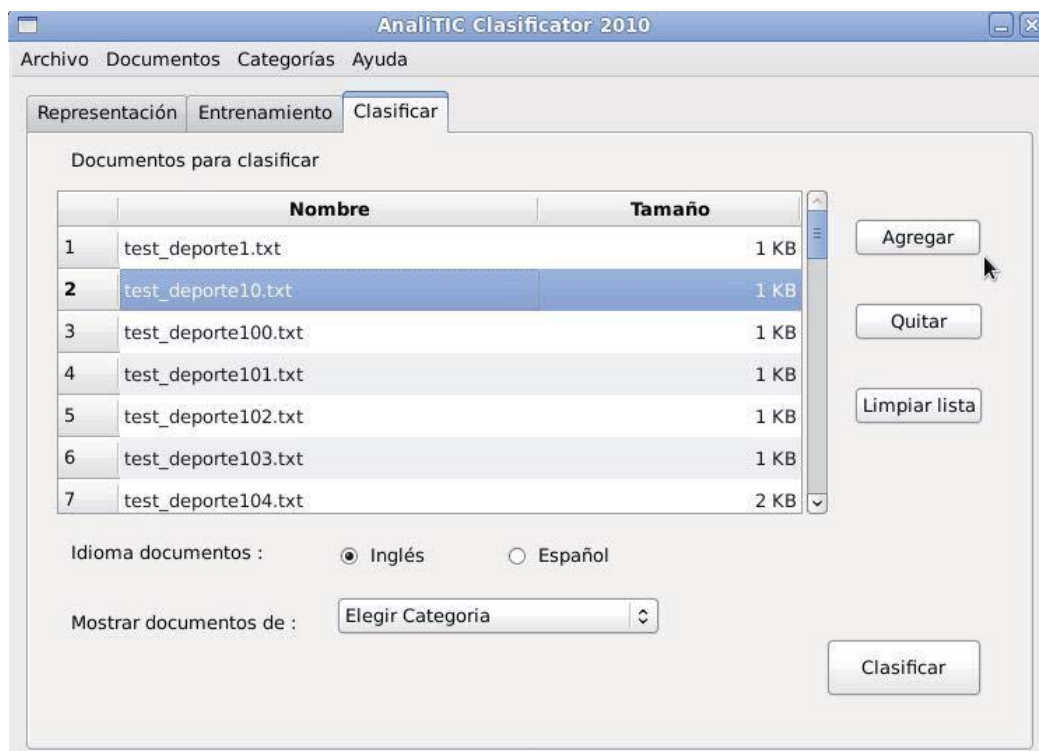


Figura 5.19 Interfaz Clasificar

En esta pantalla se clasifican los documentos. Para esto primero se deben cargar, eligiéndolos de un directorio, luego se debe elegir si los documentos serán clasificados en una categoría en inglés o español y finalmente se elige la categoría.

Al presionar el botón de clasificar se efectuará la clasificación de los documentos utilizando el modelo asociado a la categoría elegida y se mostrará una pantalla exponiendo el contenido de los documentos que fueron clasificados pertenecientes a la categoría antes elegida (documentos de interés) así como los documentos que no pertenecen a esa categoría (otros documentos).

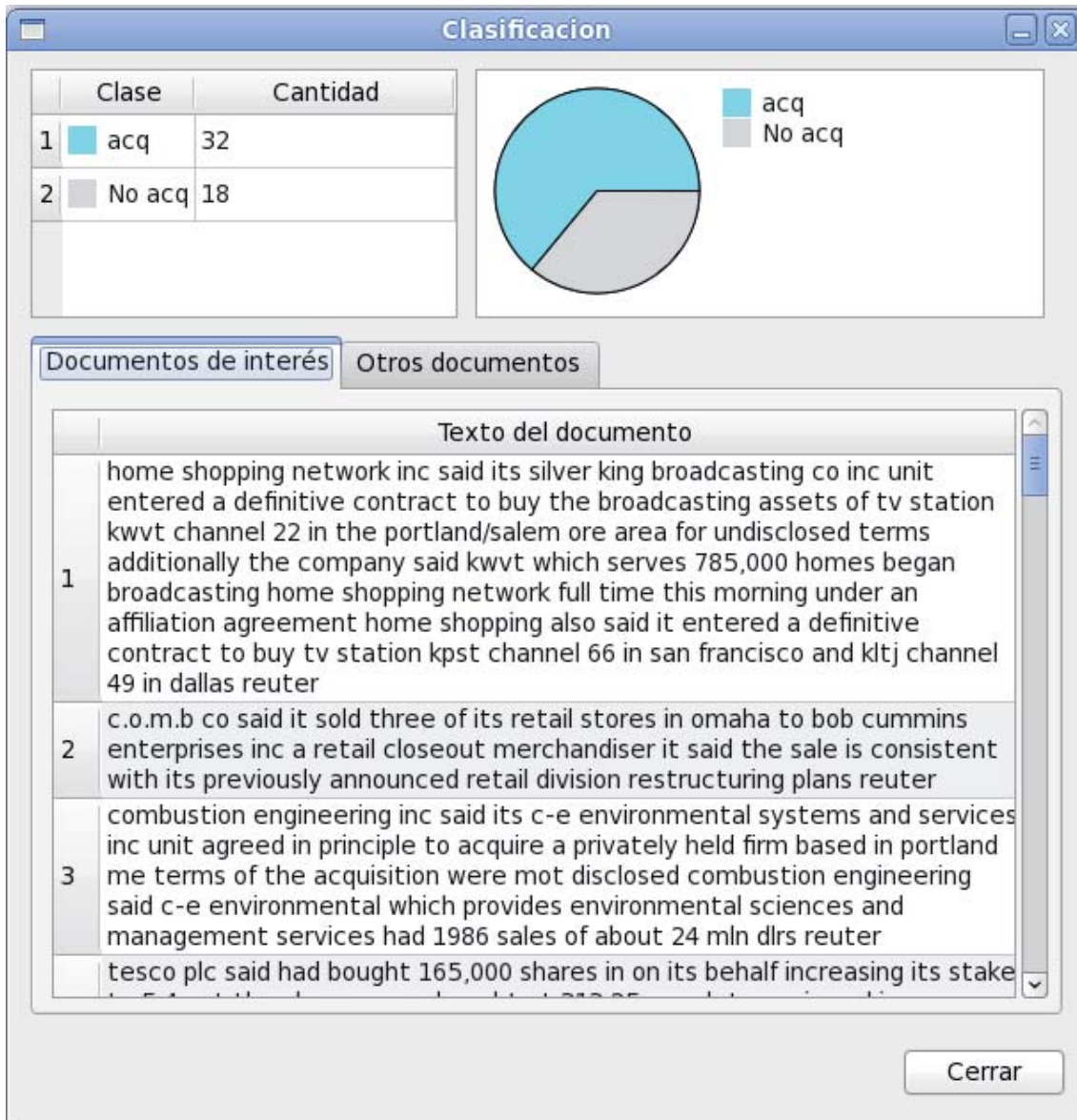


Figura 5.20 Pantalla Clasificación final

5.5.4 Interfaz Editar Categorías

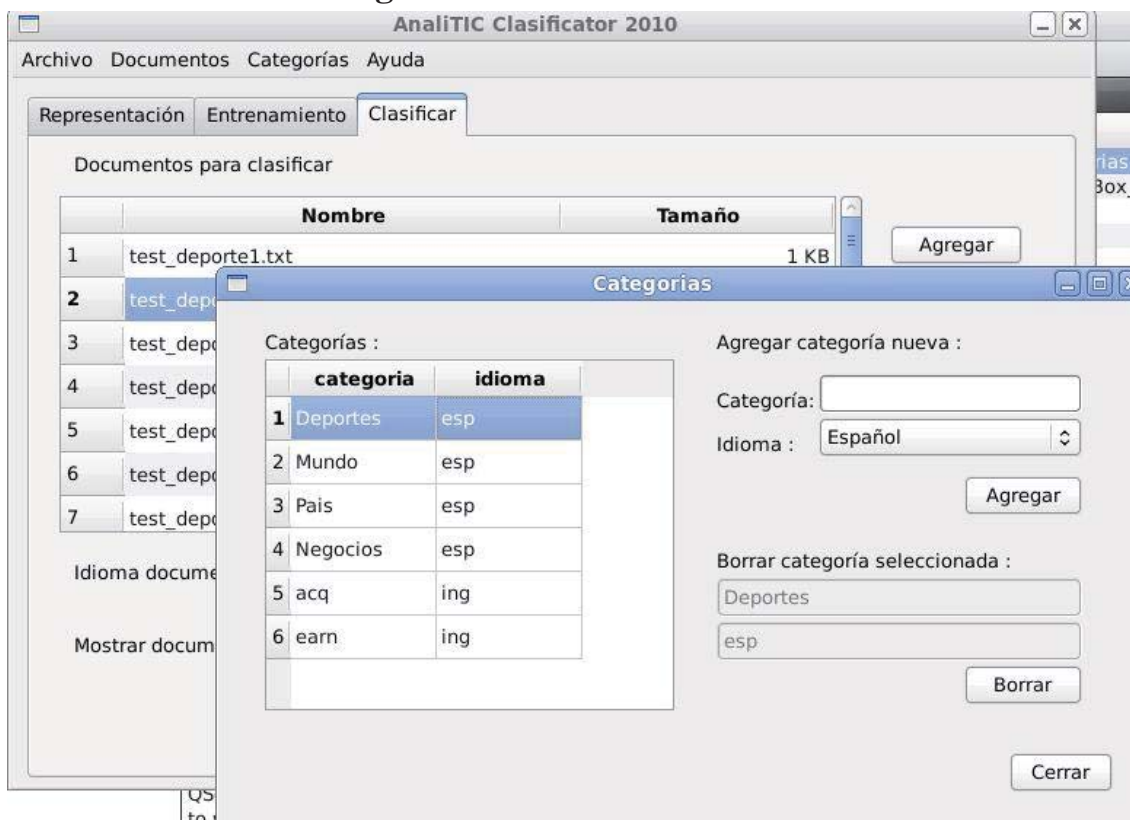


Figura 5.21 Interfaz Editar Categorías

En esta pantalla, a la que se puede acceder en cualquier momento mediante la barra de menú, se pueden insertar nuevas categorías eligiendo el idioma al que pertenecen y de igual manera se pueden borrar categorías.

5.5.5 Interfaz Editar Documentos

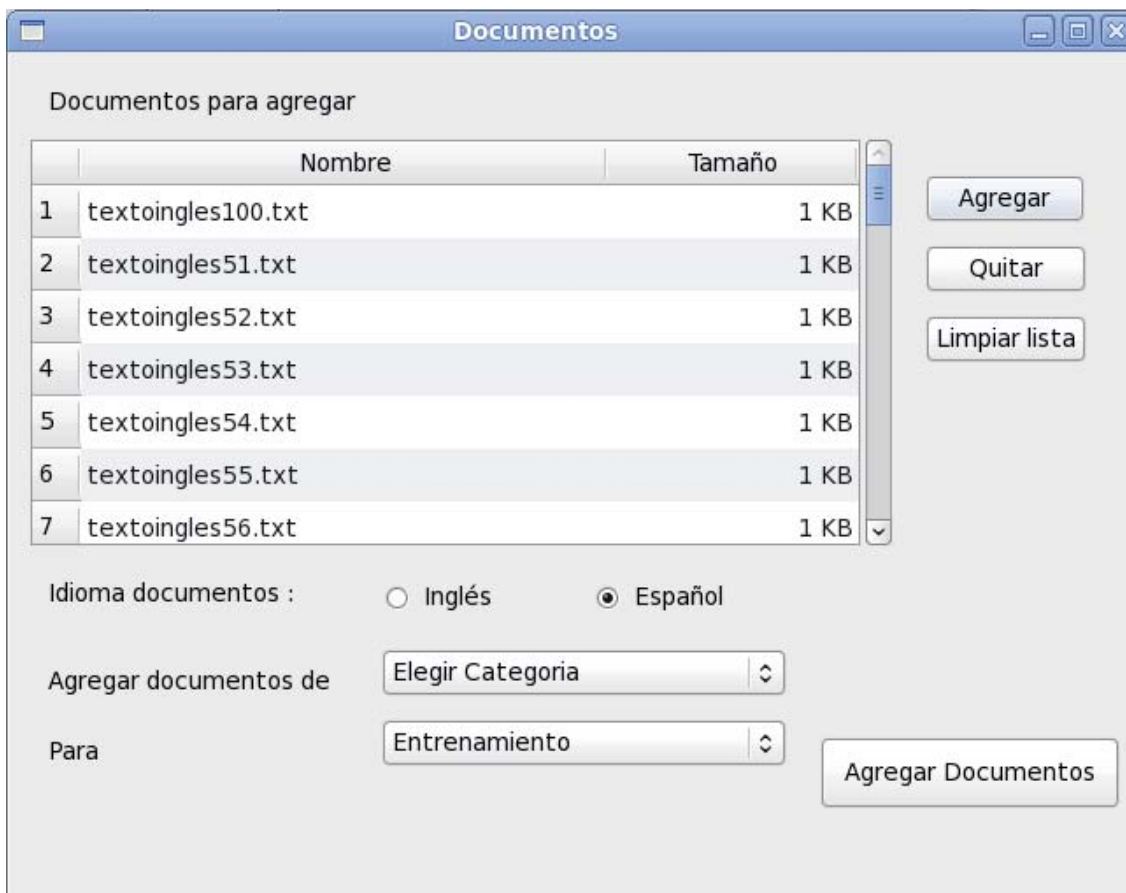


Figura 5.22 Interfaz Editar Documentos

En esta pantalla, a la que se puede acceder en cualquier momento mediante la barra de menú, se pueden insertar nuevos documentos eligiendo a qué categoría e idioma pertenecen y si serán destinados para ser documentos de entrenamiento o documentos de prueba para ser usados en la evaluación del clasificador.

5.5.6 Diagrama de estado de las interfaces del software

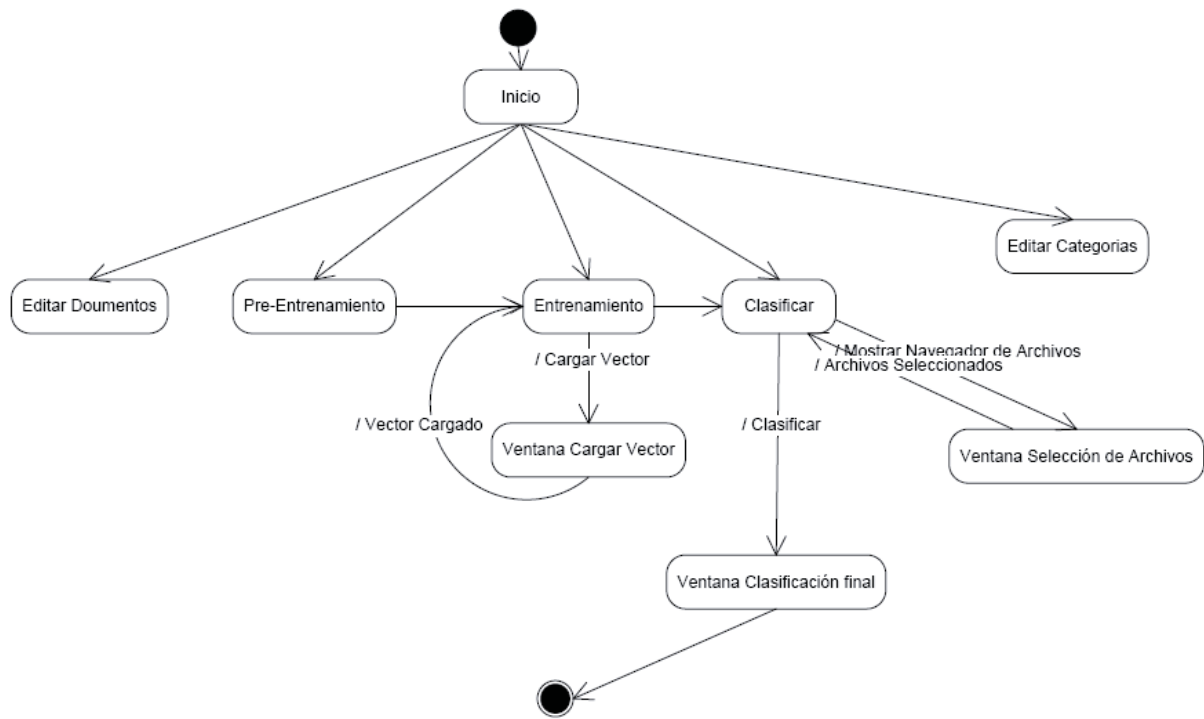


Figura 5.23 Diagrama de estado de las interfaces del software

6 Caso de estudio utilizando el software construido

Aprovechando las funcionalidades del software construido, la inclusión de más representaciones vectoriales de documentos (Representación binaria y Representación TF RF) y la sugerencia de parámetros de kernel, se realizó otra prueba para comprobar la influencia que tiene la elección de distintos documentos como documentos de entrenamiento y/o documentos de prueba.

Para realizar la comparación se seleccionaron 1000 documentos del set de datos de Reuters y La Tercera, seleccionando dos tercios de los documentos (667) para entrenamiento y un tercio (333) como documentos de prueba para evaluar, proporción sugerida mayoritariamente en la literatura para este tipo de investigación.

Como parte del diseño experimental se hace necesario obtener establecer que tan confiables son los resultados obtenidos, para ello la selección de conjuntos de entrenamiento y pruebas se hizo aleatoriamente en cinco ocasiones sobre el conjunto de datos, para así obtener distintas selecciones. Luego se procedió a entrenar y probar la máquina de aprendizaje siguiendo los mismos pasos de las pruebas anteriores y finalmente se presenta un análisis estadístico sobre los resultados.

Para cada selección el tamaño del vector de palabra fue diferente como era esperable quedando estos con los valores de; 7123 palabras para la primera selección; 6747 palabras para la segunda selección; 6831 para la tercera selección; 7006 para la cuarta selección y 6744 para la quinta, en el caso de Reuters.

Para “La Tercera” el tamaño de los vectores fueron los siguientes: 14436 para la primera selección, 12985 para la segunda selección, 14311 para la tercera selección, 14555 para la cuarta selección y finalmente 14352 para la quinta selección.

A continuación se presentan los resultados de la medida F_1 descrita en 3.4.6.2.5 para cada selección de documentos diferenciados por representación.

Tabla 6.1 Resultados F_1 para Reuters usando SVM

	1°	2°	3°	4°	5°
Binaria	96,47%	98,52%	95,91%	96,43%	99,11%
Frecuencia	97,94%	98,51%	98,24%	95,91%	97,62%
TF	97,06%	98,81%	97,93%	96,49%	98,24%
TF IDF	94,43%	94,49%	97,31%	94,74%	95,55%
TF RF	95,88%	97,89%	97,04%	96,43%	97,94%

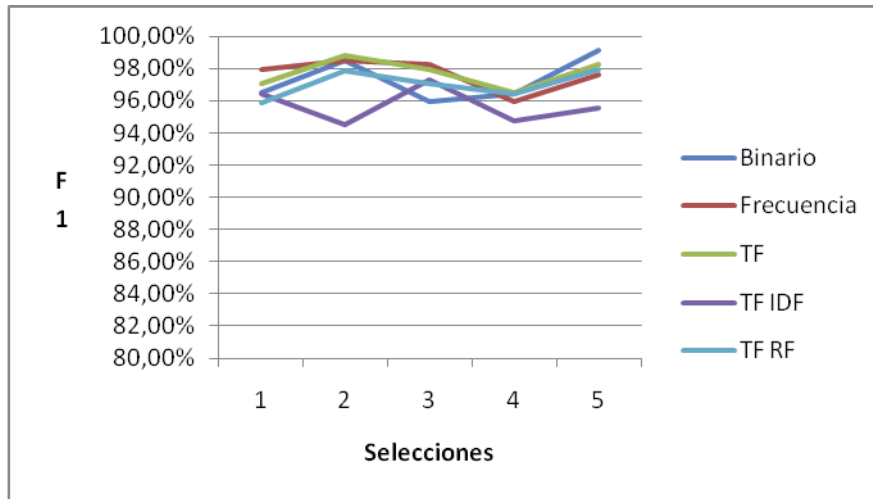


Figura 6.1 Resultados F_1 para Reuters usando SVM

Tabla 6.2 Resultados F_1 para Reuters usando Naive Bayes

	1°	2°	3°	4°	5°
Binaria	88,83%	92,78%	92,78%	88,83%	88,83%
Frecuencia	92,78%	92,78%	92,78%	88,83%	88,83%
TF	88,83%	92,78%	92,78%	88,83%	88,83%
TF IDF	88,83%	92,78%	92,78%	88,83%	88,83%
TF RF	88,83%	92,78%	92,78%	88,83%	88,83%

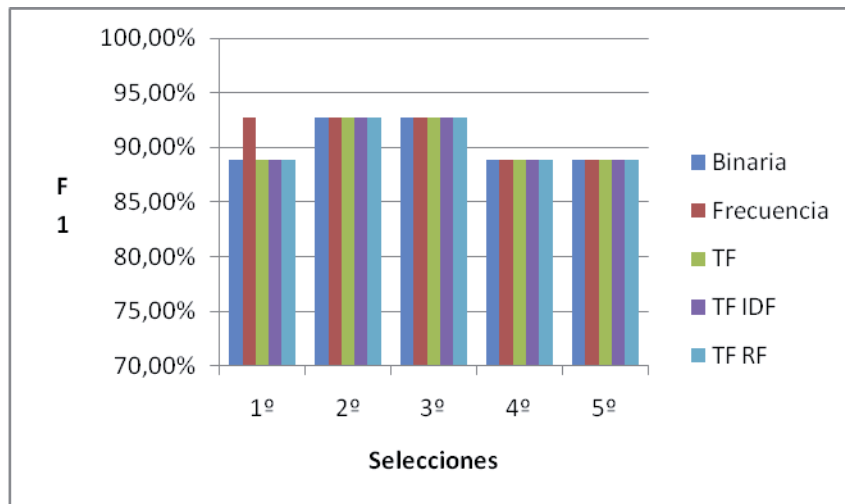


Figura 6.2 Resultados F_1 para Reuters usando Naive Bayes

Tabla 6.3 Resultados F_1 para La Tercera usando SVM

	1°	2°	3°	4°	5°
Binaria	97,55%	97,60%	99,70%	97,86%	96,91%
Frecuencia	98,79%	98,20%	98,80%	97,86%	97,87%
TF	97,89%	97,90%	98,50%	98,80%	97,90%
TF IDF	97,90%	97,92%	98,21%	98,50%	98,19%
TF RF	96,21%	96,76%	98,52%	97,92%	97,90%

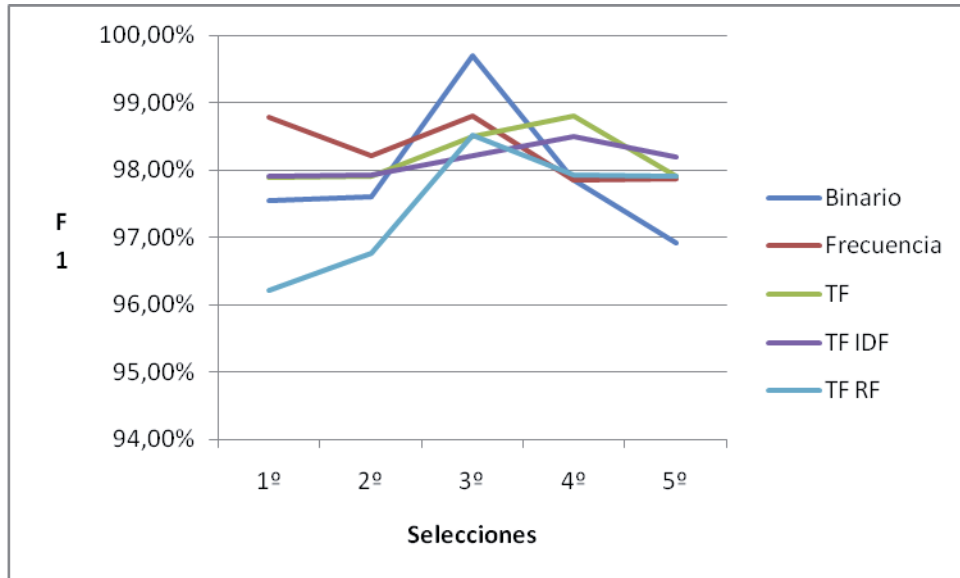


Figura 6.3 Resultados F_1 para La Tercera usando SVM

Tabla 6.4 Resultados F_1 para La Tercera usando Naive Bayes

	1°	2°	3°	4°	5°
Binaria	81,27%	74,39%	84,77%	82,27%	80,68%
Frecuencia	81,27%	74,39%	84,77%	82,06%	80,87%
TF	81,27%	74,39%	82,47%	82,06%	80,87%
TF IDF	81,27%	74,39%	84,77%	82,06%	80,87%
TF RF	81,27%	74,39%	84,77%	82,06%	80,87%

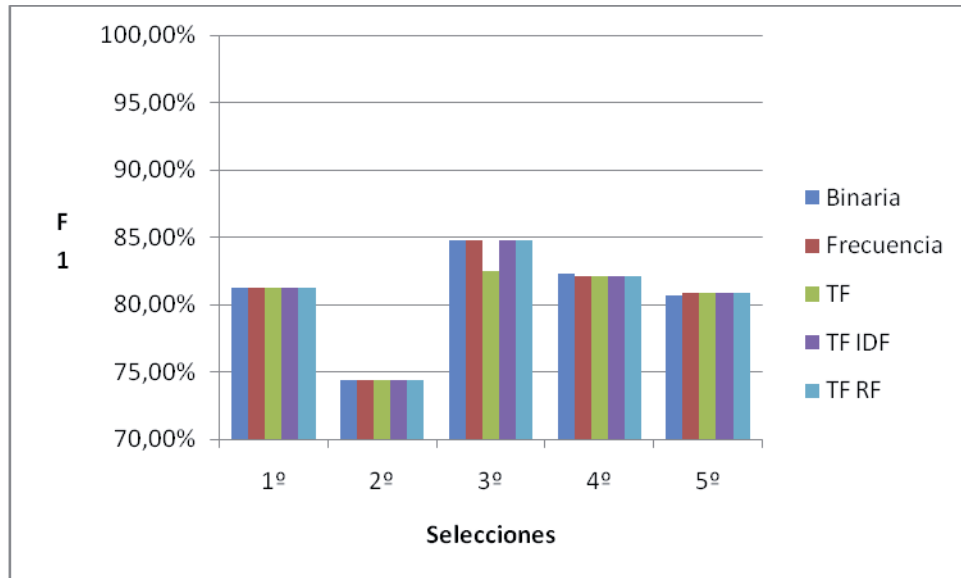


Figura 6.4 Resultados F_1 para La Tercera usando Naive Bayes

Finalmente para presentar la distribución del F_1 obtenido de la clasificación efectuada bajo las distintas selecciones y comparar las cinco representaciones estudiadas se utiliza un diagrama de boxplot.

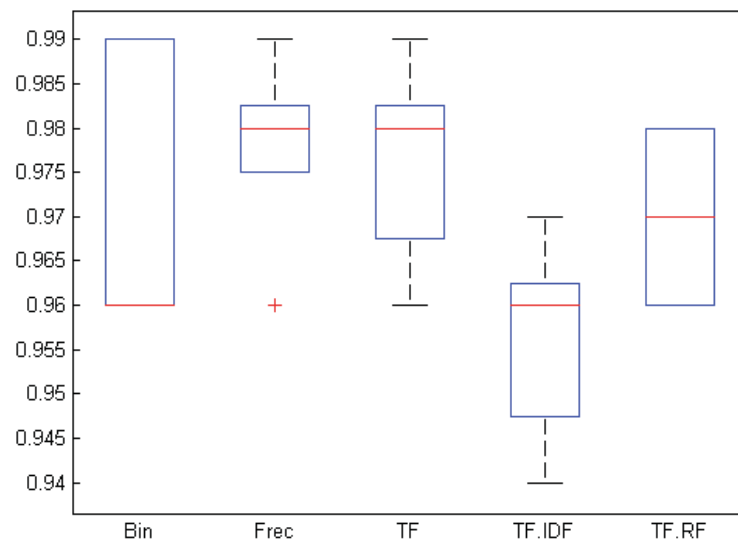


Figura 6.5 Diagrama de Boxplot F_1 para Reuters usando SVM

En el caso de SVM para Reuters se aprecia una diferencia en la distribución de para las selecciones. En lo concerniente al desempeño de la clasificación la tendencia muestra un elevado F_1 , sobre el 95%, con la excepción de TF*IDF que obtiene resultados bajo esa cota.

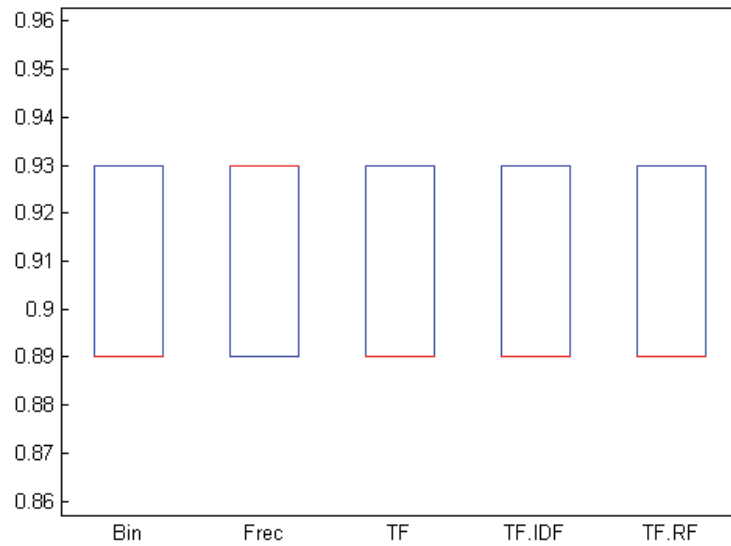


Figura 6.6 Diagrama de Boxplot F_1 para Reuters usando Bayes

Para el algoritmo de Bayes Ingenuo sobre Reuters se puede apreciar un comportamiento similar, tanto en la distribución de los datos como en el desempeño del clasificador.

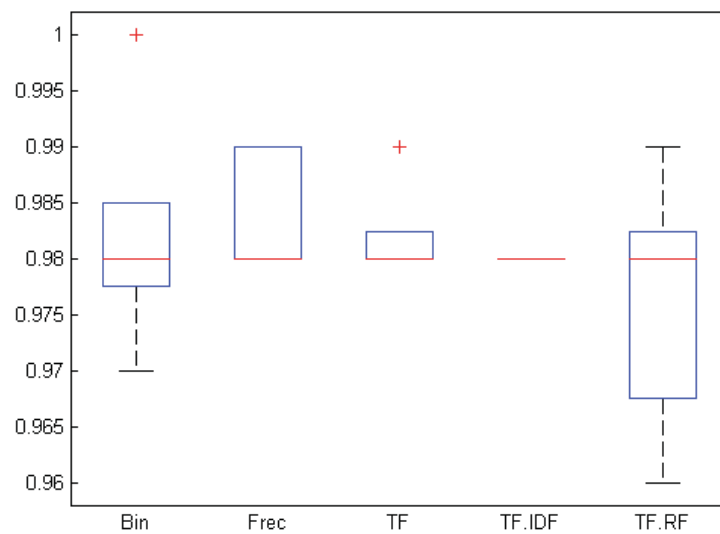


Figura 6.7 Diagrama de Boxplot F_1 para La Tercera usando SVM

En el conjunto de datos de La Tercera usando el clasificador SVM (Figura 6.7), se puede apreciar un desempeño regular para las cinco representaciones escogidas, todas sobre el

95%, sobre la distribución de los resultados para cada representación, la binaria y TF*RF presentan menor simetría que el resto de las representaciones que muestran una simetría regular. Para el clasificador Bayes Ingenuo, se observa en la figura 6.8 una clara asimetría en la distribución de los resultados para todas las representaciones probadas y un resultado similar en el desempeño del clasificador con la observación de la representación TF que a pesar de tener mediana similar, presenta un valor máximo de F_1 menor comparado con el resto de las representaciones.

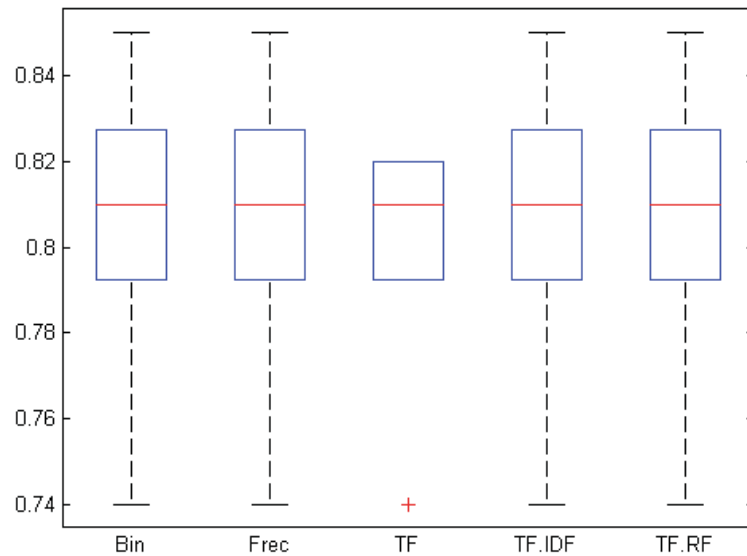


Figura 6.8 Diagrama de Boxplot F_1 para La Tercera usando Bayes

7 Conclusiones

Mediante la realización de este proyecto se pudo vislumbrar que la problemática de la clasificación automática de texto es transversal a distintos campos organizacionales y se hace necesario enfrentarla, con el fin de tener acceso fácil a la información.

La clasificación automática de texto puede conseguirse mediante distintas técnicas. Pero el uso de máquinas de aprendizaje se ha convertido en el más utilizado, por lo que se han revisado distintos métodos, poniendo énfasis en los más populares por su robustez y resultados.

Las Máquinas de Soporte Vectorial (SVM) y Naive Bayes (Bayes Ingenuo) presentan resultados equiparables, según la literatura, a los conseguidos por clasificadores inteligentes, es decir, humanos calificados. Como principales ventajas de destacan que las SVM puede trabajar con vectores de alta dimensionalidad y Bayes destaca por su rapidez de ejecución.

En relación a los objetivos planteados, se ha cumplido con la investigación de las distintas técnicas de representación de texto, reducción de dimensionalidad y máquinas de aprendizaje, además se pudo comparar los resultados de estas técnicas mediante la realización de un caso de estudio, para así determinar la elección de las mejores técnicas que se adecuan al problema de la clasificación automática de texto. Se ha definido la metodología a usar para el desarrollo del sistema clasificador, con lo cual se ha modelado en su totalidad y llevado a implementación.

Finalmente se ha utilizado el sistema construido para la realización de las pruebas finales de ambas máquinas de aprendizaje sobre un benchmark utilizado para los fines de la clasificación automática de texto (Reuters), aplicando distintas técnicas para su representación, además se estableció un diseño experimental que permitió encontrar un grado de significancia estadística a los resultados obtenidos.

Como trabajo futuro se propone profundizar en técnicas de reducción de dimensionalidad, propias de la lingüística aparte de la selección de palabras comunes, como también técnicas de selección de características propias del proceso de clasificación con máquinas de aprendizaje que sean efectivas en el caso puntual de la clasificación de texto. Teniendo como objetivo mejorar los resultados y construir un sistema más robusto que se encargue de resolver el problema propuesto en este trabajo de título.

8 Referencias

- [1] Alpaydin, Ethem, Introduction to Machine Learning, MIT 2004.
- [2] Baeza-Yates Ricardo, Ribeiro-Neto Berthier, Modern Information Retrieval. Addison-Wesley, Wokingham, UK, 1999.
- [3] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin, A Practical Guide to Support Vector Classification, 2009.
- [4] Chih-Wei Hsu and Chih-Jen Lin, A Comparison of Methods for Multi-class Support Vector Machines, 2002
- [5] Gómez Hidalgo, J.M. Clasificación de texto con adversario: técnicas de clasificación y filtrado aplicadas a la detección de spam en la Web. Curso de Tecnologías Lingüísticas: Técnicas de extracción y visualización de información: aplicación en la construcción de portales especializados. Fundación Duques de Soria (Soria), dirigido por la Prof.^a Felisa Verdejo, de la Universidad Nacional de Educación a Distancia, 7 al 11 de julio de 2008.
- [6] Gómez Hidalgo, J.M., Cortizo Pérez, J.C., Puertas Sanz, E., Buenaga Rodríguez, M. de. Experimentos en indexación conceptual para la categorización de texto. In Gutiérrez, J.M., Martínez, J.J., Isaías, P. (Eds) Actas de la Conferencia Ibero-Americana WWW/Internet 2004, Madrid, Spain, October, 7-8, pp. 251-258 2004.
- [7] Gómez Hidalgo, J.M. Tutorial on Text Representation for Automatic Text Categorization. Eleventh Conference of the European Chapter of the Association for Computational Linguistics, Budapest, Hungary, 12-17 April 2003
- [8] Joachims Thorsten, Transductive Inference for Text Classification using Support Vector Machines 1999.
- [9] Joachims, Thorsten, Learning to Classify Text using Support Vector Machines. Method, Theory and Algorithms 2002.
- [10] Kjersti, Aas, Line, Eikvil, Text Categorisation: A Survey 1999.
- [11] Manning, Christopher, D., Prabhakar Raghavan, Hinrich Schütze Introduction to Information Retrieval, Cambridge University Press. 2008.
- [12] Meisner, Eric, A Naïve Bayes Classifier Example 2003.
- [13] Mitchell, Tom, Machine Learning Mc-Graw Hill, 1997
- [14] Pressman, Roger, Ingeniería de Software, Un Enfoque Práctico Quinta Edición
- [15] Sebastiani, Fabrizio, Machine learning in automated text categorization. ACM Computing Surveys, Vol. 34, No. 1, pp. 1-47, 2002.

[16] Sung, San-Yuan. A Comparative Study on Term Weighting Schemes for Text Categorization. 2005

[17] Venegas, René. Clasificación de textos académicos en función de su contenido léxico-semántico. Rev. signos, 2007, vol.40, no.63, p.239-271. ISSN 0718-0934

[18] Yang Y., Petersen J. A Comparative Study on Feature Selection in Text Categorization 1997.