UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

DISEÑO DE UN NUEVO SISTEMA DE VIGILANCIA DE DOMINIOS DE INTERNET

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRICISTA

RODRIGO ANDRÉS ZUMAETA MORALES

PROFESOR GUÍA:
PABLO ESTÉVEZ

MIEMBROS DE LA COMISIÓN:
HÉCTOR AGUSTO
JORGE LÓPEZ

SANTIAGO DE CHILE
AGOSTO 2007

RESUMEN DE LA MEMORIA
PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELECTRICISTA
POR: RODRIGO A. ZUMAETA M.
FECHA: 20/08/07
PROF. GUÍA: Sr. PABLO ESTEVEZ

"DISEÑO DE UN NUEVO SISTEMA DE VIGILANCIA DE DOMINIOS DE INTERNET"

El objetivo general de esta memoria es presentar una solución para detectar intentos de plagio de dominios de internet, mejorando los resultados entregados por un sistema preexistente.

El trabajo fue realizado para una empresa que se encarga de registrar y proteger dominios de internet para firmas comerciales, con el problema de que el sistema utilizado en la protección entregaba cantidades excesivas de falsas alarmas. Este sistema resulta poco práctico en términos económicos, debido al costo que representa el descarte manual de este gran número de falsas alarmas.

Se presentó una solución basada en el modelamiento del problema y aplicación de algoritmos clasificadores entrenables.

El trabajo incluyó fases de modelamiento o traducción del problema, diseño, implementación, construcción de set de datos para entrenar y validar los algoritmos, y pruebas con datos reales para comparar los resultados de la solución propuesta con el sistema que se pretendía superar.

El resultado es una solución que reduce la cantidad de alarmas reportadas diariamente a cerca de un tercio (y las *falsas* alarmas al 15%) de las obtenidas con el sistema antiguo y que no incrementa la cantidad de alarmas reales no reportadas.

Se concluye que el objetivo perseguido se cumplió de modo exitoso, superando ampliamente los resultados del sistema antiguo y entregando a la empresa una alternativa más eficaz y rentable.

Índice

1 Introducción

En el ámbito corporativo, la protección de elementos representativos (como marcas y logotipos) ha tenido una importancia fundamental. A nivel de marketing, las grandes empresas buscan lograr una diferenciación mediante la identificación de estos elementos representativos, logrando que el cliente reconozca a la empresa y genere, muchas veces, una relación de lealtad con la misma y planteando, al mismo tiempo, dificultades y barreras de entrada a potenciales competidores que pretendan quitar terreno a estas grandes marcas establecidas.

El plagio o "piratería de marcas" es una amenaza a la identidad que logran estas grandes empresas y a su identificación con la clientela, y es causa de millonarios y extensos procesos judiciales. El rápido crecimiento de Internet como plataforma de transmisión y divulgación de contenidos y de intercambio comercial, ha agregado una nueva variante al problema previamente descrito: la necesidad de registrar y proteger dominios de Internet por parte de grandes y medianas empresas.

En efecto, existen personas que han hecho negocios registrando dominios de Internet y luego vendiéndolos a un mayor precio a las empresas que puedan estar interesadas en ellos, y se han dado bullados casos de grandes corporaciones multinacionales que han llegado a negociar o han llevado a juicio casos de este tipo (por ejemplo, el caso Microsoft vs. MikeRoweSoft, descrito en [13]).

Un juicio para recuperar un dominio de Internet puede ser costoso y lento. Este último problema es especialmente relevante considerando los tiempos que se manejan en la Internet, y el hecho de que la velocidad en la propagación de la información es una de sus características principales. Similares problemas puede tener la negociación con el dueño del registro del dominio.

Por lo anterior, la solución más idónea para una empresa es detener la piratería de dominios antes de que esta se lleve a cabo, o, en otras palabras, adelantarse a los piratas en el registro de dominios. Es, sin embargo, en extremo complejo prevenir todas las copias exactas y variantes (fonéticas, por ejemplo) que se puedan registrar.

1.1 Planteamiento del problema

En el caso de nuestro país, es "NIC Chile" el organismo encargado de registrar dominios de Internet a nivel nacional (dominios *.cl*), esta institución cuenta con mecanismos de arbitraje que permiten solucionar disputas al presentarse más de una solicitud por un mismo dominio.

De este modo, una solución frecuentemente utilizada por grandes empresas es poner sus dominios de interés en manos de un especialista que vigila las nuevas solicitudes de dominios registradas en NIC. El especialista tendrá la función de buscar si alguno de los dominios solicitados puede corresponder a alguna forma de plagio de alguno de los dominios que se le ha encargado proteger, y en este caso, efectuar simultáneamente el registro para llevar el caso a arbitraje.

El problema para un especialista en vigilancia de dominios en Chile es que .en NIC cada día hábil se registra un número promedio no menor a las 370 solicitudes de dominios. A eso, se debe agregar que su clientela, idealmente, consiste en un número importante de empresas que pretendan proteger uno o varios dominios. Esto puede derivar en un número de decenas de miles de dominios a proteger, los cuales se deben comparar con 370 registros diarios.

Se hace necesario, considerando lo anteriormente planteado, contar con

algoritmos eficaces para detectar plagios de dominios, entendiendo por "eficacia" la capacidad del algoritmo de lograr bajos los porcentajes de falsos positivos y de falsos negativos en el reporte. Un algoritmo ideal es el que reporta la totalidad de similitudes relevantes diariamente, sin reportar ninguna similitud relevante: es decir, da aviso (exclusivamente) de todos los verdaderos positivos.

1.2 Objetivos

En el caso particular de esta memoria, el trabajo se hizo para una empresa especialista en la protección de dominios, que contaba con un algoritmo de vigilancia. El objetivo era mejorar los resultados obtenidos por este algoritmo, especialmente en cuanto a disminuir el porcentaje de similitudes irrelevantes entregadas, que era, según la evaluación hecha por la empresa, su principal defecto.

Se propone, en este trabajo, una solución basada en el modelamiento del problema, presentándolo como un problema de tipo numérico y la posterior clasificación de los resultados mediante el uso de algoritmos de aprendizaje.

1.2.1 Objetivo General

El objetivo general perseguido en el trabajo corresponde, entonces, al diseño de un algoritmo de protección de dominios de Internet de alta eficacia, es decir, que sea capaz de distinguir los intentos de plagios de dominios cometiendo relativamente pocos errores (ver más precisión sobre los niveles de error deseados en la sección de objetivos específicos).

1.2.2 Objetivos Específicos

Detallando lo anterior, los objetivos específicos del trabajo son:

- Obtener, *mediante modelamiento y aplicación de algoritmos de aprendizaje*, un mayor porcentaje de éxito en el reporte de similitudes irrelevantes que el sistema existente en la empresa que solicitó el trabajo. En efecto, el requisito impuesto consiste en reducir las alarmas arrojadas, en lo posible, a la mitad.
- Al mismo tiempo, evitar el aumento de error obtenido en cuanto al reporte de similitudes relevantes. Es decir, reducir a la mitad la cantidad de similitudes irrelevantes entregadas, pero intentando que no aumente el número de similitudes relevantes no reportadas.

Para lograr estos objetivos, se deben considerar diversos pasos metodológicos, que pueden considerarse objetivos de carácter intermedio:

- Observar el sistema previo de detección de plagios usado por la compañía. Se considera de primera importancia comprender y analizar su funcionamiento, y luego de esto realizar un diseño distinto (que puede o no estar basado en el método anterior) que mejore los resultados, permitiendo alcanzar los objetivos planteados anteriormente.
- Modelar el problema existente. Realizar una formulación del problema que permita una resolución algorítmica que apunte a la obtención de los resultados pretendidos.
- Búsqueda, diseño e integración de los algoritmos que permitan resolver el problema modelado.
- Construir sets de datos que permitan realizar las pruebas pertinentes. Como se mencionó anteriormente, se pretende una solución basada en algoritmos de aprendizaje. Para esto, es necesario construir conjuntos de datos que permitan entrenar, validar y probar los algoritmos.

- Finalmente, se debe señalar que se pretende realizar un estudio comparativo entre el nuevo diseño y el método antiguo de la empresa, probando ambos *con datos reales obtenidos diariamente* (no solamente con conjuntos de datos previamente construidos para realización de pruebas) y demostrar las mejoras obtenidas con el diseño propuesto con este método, de modo concluyente.

1.3 Alcances del Trabajo

Los alcances del presente trabajo, son:

- Realizar una descripción tanto del algoritmo previo, el que se pretende mejorar, como de la solución implementada (el nuevo diseño realizado).
- Realizar una descripción cualitativa de los procesos de diseño y pruebas correspondientes.
- Analizar cuantitativamente los resultados obtenidos en comparación con la situación a mejorar.
- Establecer conclusiones sobre el trabajo realizado y el grado éxito del mismo, basado en los análisis comparativos presentados.

1.4 Importancia del trabajo realizado.

La relevancia del trabajo realizado para la empresa solicitante es clara: la obtención de un sistema de protección de dominios con un funcionamiento más eficaz, el cual permitiría optimizar el tiempo, aumentar la productividad y redestinar recursos.

Al iniciar el trabajo, la empresa contaba con un algoritmo de protección de dominios que reportaba un promedio de 3.380 similitudes diarias. De este número de similitudes, un promedio cercano al 20% correspondía a similitudes

relevantes, que valía la pena reportar a los clientes.

El descarte de similitudes irrelevantes era realizado manualmente por tres personas de la empresa, las cuales destinaban a esta labor un total de, aproximadamente, seis horas diarias. Esto implica que tres personas de la empresa utilizaban, al menos, dos tercios de su jornada laboral en relizar solamente este trabajo de descarte, lo cual significaba un obstáculo para destinar a estas personas a otro tipo de labores y de mejorar, de este modo, el rendimiento y eficiencia de la empresa.

El trabajo realizado tiene importancia práctica en este sentido, permitiendo a la empresa un uso más eficiente de su personal y sus recursos. Además de lo anterior, se presenta un aporte en un sentido más conceptual, al plantearse una solución nueva y original en la protección de dominios de internet y en el modelamiento del problema de similitudes de strings aplicado a la detección de plagios de dominios de internet.

En este sentido, se ofrece una solución diseñada particularmente para este problema, nacida de la observación de las complejidades específicas que esta aplicación presenta, y que no se ha planteado previamente en publicaciones.

2 Antecedentes.

2.1 Situación preexistente: el algoritmo antiguo

El algoritmo que se pretende mejorar con el siguiente trabajo está programado con métodos tradicionales. Este algoritmo se puede describir de modo estructurado del siguiente modo:

1. *Obtener la lista de nuevas solicitudes registradas en NIC Chile y guardarlas en una tabla.*
2. *Se toma un String protegido*
3. *Se toma un String de la tabla de nuevas solicitudes.*
4. *Se reducen ambos strings fonéticamente, es decir, se hacen equivalencias entre los caracteres y las secuencias de caracteres que, fonéticamente, suenan iguales o parecidas ("co" es equivalente cono "ko", "ce" es equivalente con "se", etcétera).*
5. *Se comparan los dos strings reducidos, de modo de ver si uno de los dos strings está contenido en el otro.*
6. *Si el String protegido está contenido en el otro, se reporta la similitud.*
7. *Si quedan strings en la tabla de nuevas solicitudes, se vuelve al punto 3, con el siguiente string de la tabla.*
8. *Si quedan strings en la lista de protegidos, se vuelve al punto 2, con el siguiente de la lista.*
9. *Finaliza.*

También es posible plantear en forma de diagrama, como se muestra en la Figura 1.

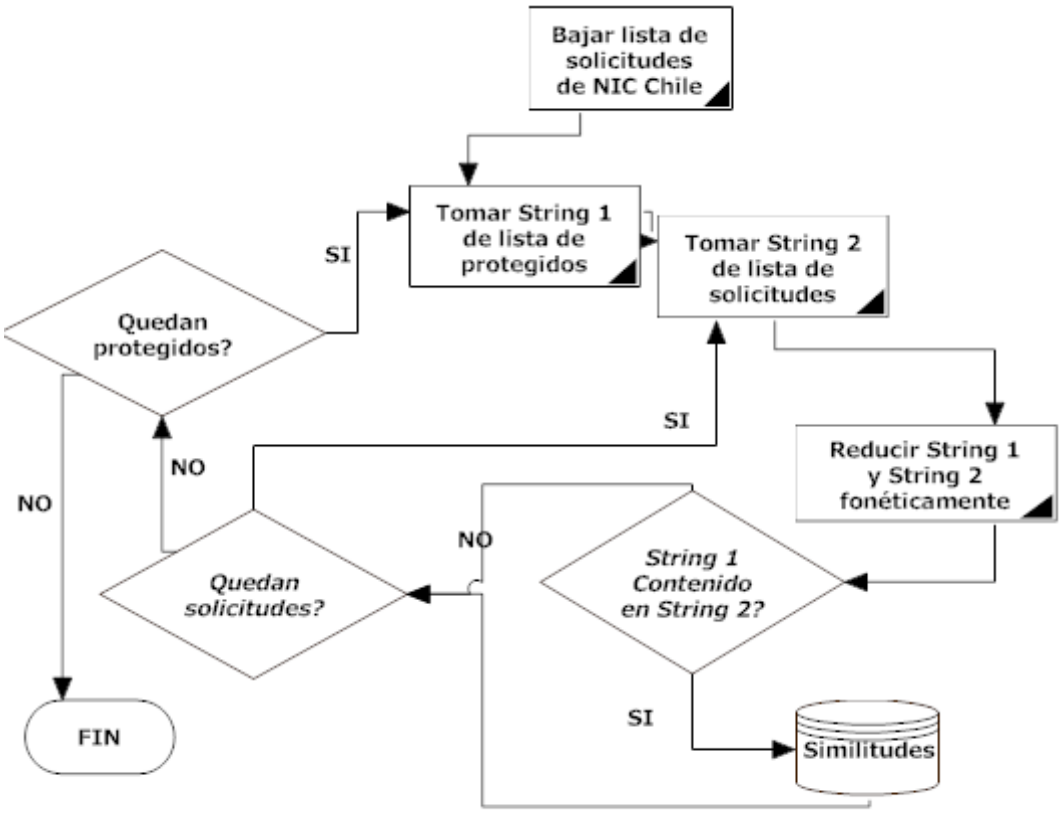


Figura 1: Diagrama de flujo del algoritmo antiguo

El promedio de similitudes reportadas diariamente con este algoritmo es 3.380[1]. De este número un promedio de 672[2] similitudes son alertadas diariamente, lo cual constituye un 19,88%.

Además del alto número de similitudes irrelevantes reportadas, el algoritmo marca como no-similitudes un número indeterminado (pero no menor) de similitudes relevantes, las cuales son descubiertas al aplicar el nuevo algoritmo y efectuar comparaciones.

1 Promedio obtenido en días hábiles entre el 26 de febrero y el 9 de marzo de 2007.
2 Idem nº 1

2.2 Antecedentes conceptuales para la solución implementada.

Como se describió *a grosso modo*, la solución escogida para el problema consiste en dos grandes pasos:

1. La traducción del problema de similitudes de strings a un problema numérico, mediante el cálculo de ciertas *características numéricas*.
2. La aplicación de un algoritmo de clasificación para separar en clases los sets de características calculados.

La primera parte del problema radica, entonces, en escoger las características a calcular, de modo de perder la menor cantidad de información útil posible para cada par de strings. La elección de estas características se basó, por una parte, en el estudio acabado de información histórica del problema en particular y por otra, en el estudio de conceptos utilizados en el campo de recuperación de información (*Information Retrieval*, o *IR*).

La segunda parte consiste en escoger un método de clasificación adecuado para clasificar los sets de características con el menor error -de cada tipo- posible. La elección de este clasificador se basó en el estudio y prueba de distintos tipos de algoritmos de aprendizaje, como redes neuronales, máquinas de soporte vectorial, tablas y árboles de decisión.

Las bases conceptuales del trabajo se exponen a continuación.

2.3 Similitud entre strings: Características cuantificables.

2.3.1 Distancia de edición o distancia de Levenshtein.

La distancia de edición es posiblemente la característica más utilizada para comparar cadenas de caracteres y consiste en una cuantificación del mínimo número de inserciones, eliminaciones y reemplazos de caracteres para transformar un string en otro (detalles al respecto se pueden encontrar en [8]).

Por ejemplo, si se consideran los strings "hola" y "boas", el cálculo de la distancia de edición para transformar el primero en el segudo sería igual a *tres*: reemplazar *h* por *b*, eliminar la *l* y agregar la *s* final.

Esta medida de similitud o de diferencia entre strings ha sido ampliamente utilizada en problemas como la detección de entradas duplicadas en aplicaciones de bases de datos (*Record Linkage*, como la aplicación descrita en [12]), comparación de cadenas de genomas (como se describe en [9]) o correctores ortográficos para lenguajes de distinta naturaleza, tal como la problemática planteada en [6].

En la sección de anexos se podrá encontrar una revisión detallada de la implementación algorítmica de la distancia de edición utilizada en el desarrollo de este trabajo.

2.3.2 Similitud basada en n-gramas.

Un n-grama es una secuencia de n caracteres consecutivos. Se han planteado métodos para comparar pares de strings basándose en la cantidad de n-gramas compartidos entre ambos. En [5] se menciona y se explica la incidencia de la separación de strings en n-gramas en el campo de la information retrieval.

De este modo, tomando un ejemplo sencillo:

Para n=2, es posible contar los *digramas* en común entre los dos strings *"dominio"* y *"domani"*.

Primero, se separan ambos strings en digramas:

- dominio: do om mi in ni io, seis digramas.
- domani: do om ma an ni, cinco digramas.

Los digramas en común entre ambos strings son: do, om, ni. Es decir, existen tres digramas en común entre *"dominio"* y *"domani"*.

2.4 Clasificación.

El problema de clasificación de sets de características se puede encarar de distintas formas. Una de ellas, es la aplicación de algoritmos tradicionales.

La aplicación de algoritmos tradicionales demanda que el programador decida umbrales para cada una de las características que componen el set, de modo de que para cualquier combinación de valores, el algoritmo decida, según si los valores de las características, a qué clase pertenece el set.

Mientras más sean las características que componen el set, el problema se vuelve más complejo, y la probabilidad de que el programador falle en determinar las mejores características posibles es muy alta.

Por este motivo, para este problema se pensó en la utilización de algoritmos de aprendizaje para realizar la clasificación: se entrena el algoritmo con un gran volumen de información histórica, de modo que este "se ajuste" según los valores de la información que reciba en el entrenamiento.

En este marco, se consideraron tres grandes vertientes dentro de los algoritmos de aprendizaje para clasificación, que se explican a continuación.

2.4.1 Redes Neuronales

Las redes neuronales son algoritmos populares dentro del área de aprendizaje de máquinas y, en particular, en problemas de reconocimiento y clasificación. Dentro de las tres vertientes a tratar, las redes neuronales constituyen, posiblemente, la más conocida dentro y fuera de círculos académicos.

Una red neuronal tiene un funcionamiento de tipo "caja negra", que recibe entradas y, mediante una función o metodología desconocida para un observador externo, calcula la salida, como se aprecia en la Figura 8.

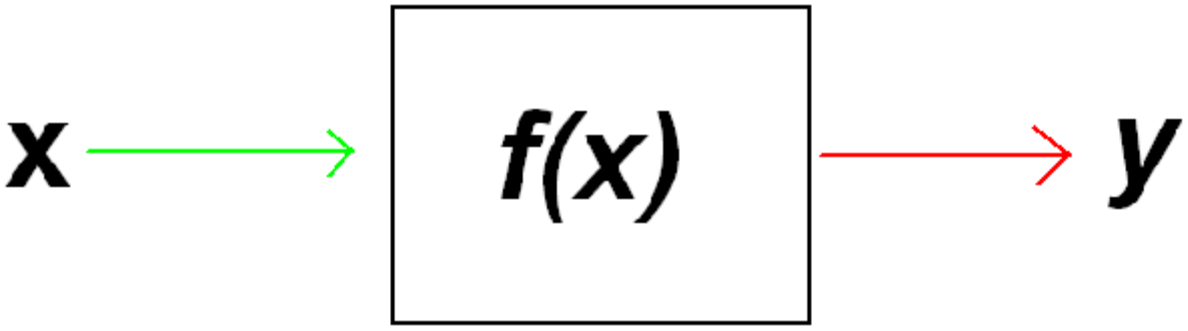


Figura 2: Algoritmo tipo "caja negra".

Como se puede extraer de [3] y de [7], las redes neuronales artificiales intentan imitar el funcionamiento de las redes neuronales naturales. El encargado de modelar y programar la red neuronal no tendrá que definir la función *f* que se aprecia en la figura, sino modelar la red neuronal, la cual implicará una determinada función.

La estructura de una neurona se puede apreciar en la Figura 3.

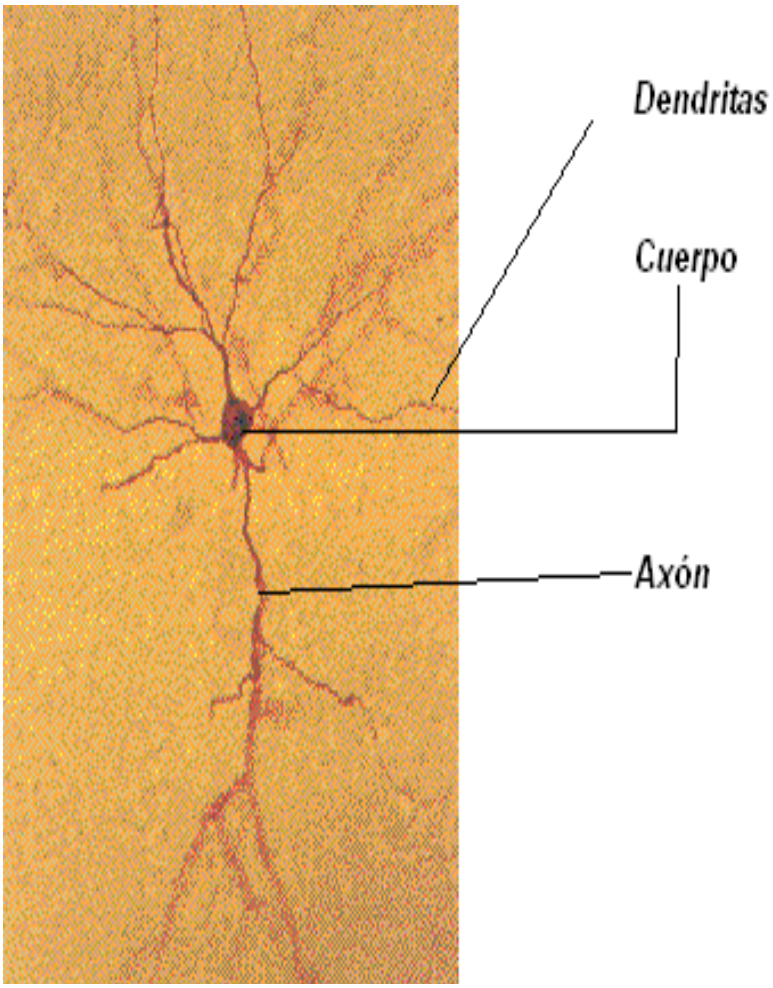


Figura 3: Partes de una neurona

Se explica en [7] que las *dendritas* son las encargadas de recibir información desde otras neuronas. El *cuerpo* se encarga de procesar información. El *axón* transmite la información hacia otras neuronas.

La interconexión de dos neuronas se llama *sinapsis*. Implica una unión entre el axón de una neurona y las dendritas de otras.

Del mismo modo, una neurona artificial se puede esquematizar como se muestra en la Figura 4.

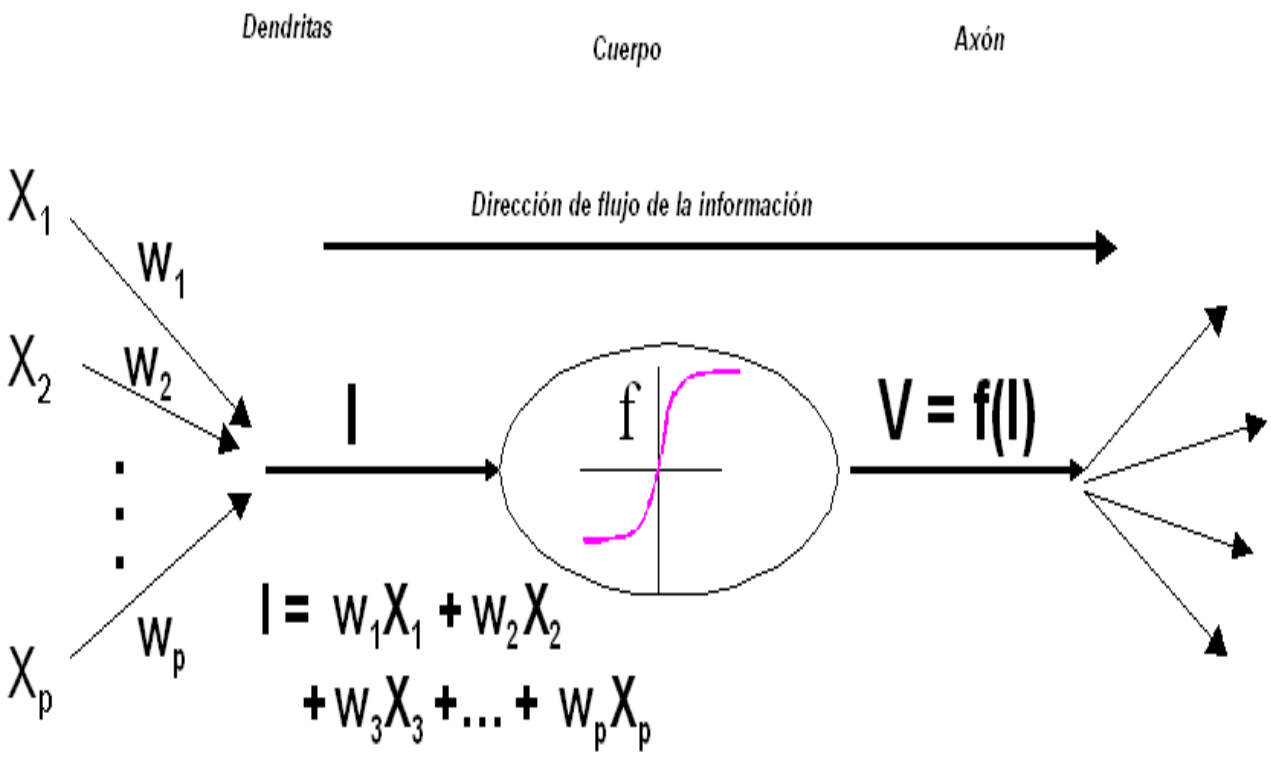


Figura 4: Neurona artificial.

La neurona de la Figura 4 recibe entradas (x) desde otras neuronas o desde el "ambiente" (entrada del sistema). Cada entrada es ingresada a través de una conexión con "peso". La entrada total corresponde a la suma ponderada -por sus respectivos pesos- de las entradas desde todas las fuentes. Una función de transferencia (función de activación) convierte la entrada en la salida. La salida va hacia otras neuronas o hacia el "ambiente" (salida del sistema).

Las neuronas en una red neuronal se organizan en "capas", como se puede apreciar en la Figura 5.

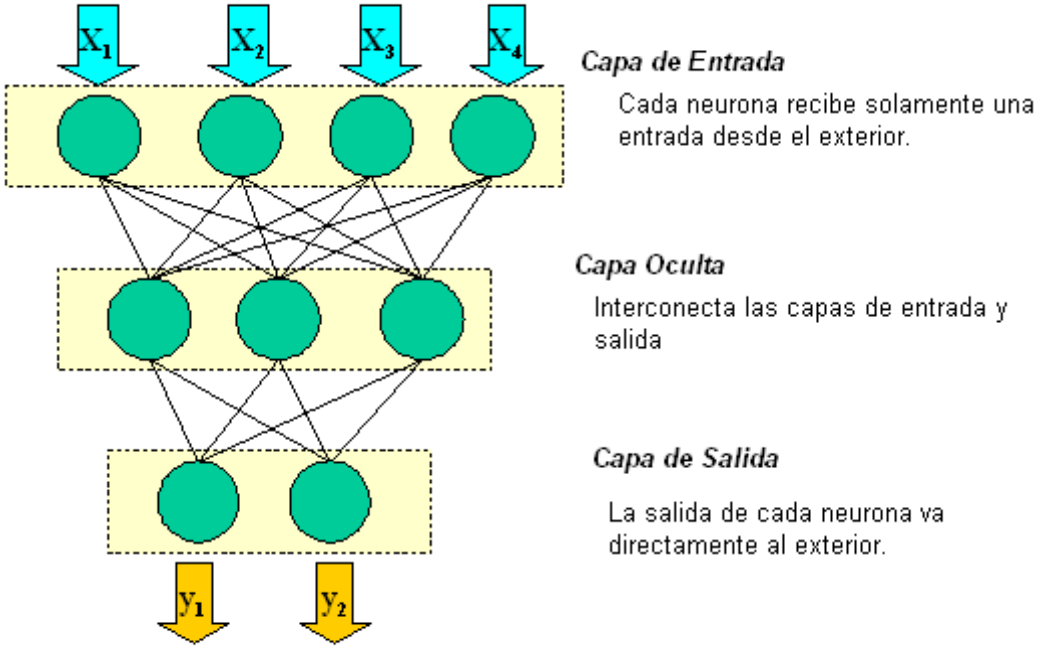


Figura 5: Red neuronal artificial.

Una red neuronal puede tener un número cualquiera de capas ocultas: cero, una o más de una.

Una red neuronal de tipo *feed-forward* cumple con las siguientes reglas:

- Las neuronas en una capa no se conectan entre sí.
- Una neurona en una capa está conectada solamente ***con las neuronas de la capa siguiente.***

Si se tienen entradas X y salidas Y de una red neuronal, el "model" de la red se refiere a la función f tal que Y=f(X).

En general, para una red neuronal la forma algebraica de la función f es muy complicada para manejar, por lo cual el modelo se caracteriza por los siguientes parámetros:

- Número de neuronas de entrada.
- Número de capas ocultas.

- Número de neuronas en cada capa oculta.
- Número de neuronas de salida.
- Pesos para todas las conexiones.

Por lo tanto, el ajustar una red neuronal se refiere a especificar los valores para estos parámetros. Algunos de estos valores serán especificados directamente por el modelador de la red, mientras otros se ajustarán en el proceso de entrenamiento (aprendizaje de la red neuronal).

Más concretamente, se deben especificar manualmente los valores para el número de neuronas de entrada, número de neuronas de salida, número de capas ocultas y número de neuronas por capa oculta.

Para entrenar la red, se debe construir un "conjunto de entrenamiento". Este conjunto consiste en un número de instancias, cada una de las cuales debe incluir un set de entradas y la salida que se espera para dichas entradas.

Los pesos se inicializan con valores aleatorios. Mediante el proceso de entrenamiento, con cada instancia del conjunto de entrenamiento, se ajustan los pesos de modo de minimizar el error entre la salida real y la salida esperada del sistema para cada instancia. Se efectúan varios ciclos de entrenamiento, de modo que el error predicho por el sistema sea "pequeño".

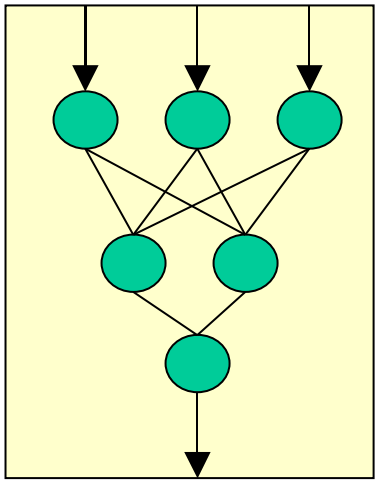


Figura 6: Ejemplo de arquitectura de red neuronal, con una capa ocutla y ocho conexiones

Por ejemplo, para la Figura 6, se definió una arquitectura de red con tres neuronas de entrada, una neurona de salida, una capa oculta y dos neuronas en la capa oculta.

Existen ocho conexiones y, por lo tanto, ocho pesos que especificar. Se define un conjunto de entrenamiento con instancias de la forma:

$$Y_i, X_{1i}, X_{2i} \ldots X_{pi}, i = 1 \ldots \mathrm{n}$$

Con X las entradas e Y la salida esperada para tales entradas.

Con un conjunto específico de pesos W_j , se obtendrá, para cada instancia del conjunto de entrenamiento, una predicción de la salida, V_i .

El error *E* está dado por la comparación entre las salidas obtenidas, V_i , y la salidas esperadas, Y_i .

De este modo, el conjunto de entrenamiento consistirá en escoger los pesos *W* tal que el error : $E = \sum (Y_{(i)} - V_i)^2$, sea mínimo.

Existe, además, el concepto de retropropagación en una red neuronal. Esto consiste en suponer que cada salida que la red predice es función de los pesos escogidos. De este modo, el error total es también función de los pesos:

$$E = \sum (Y_{(i)} - V_i)^2 = \sum (Y_{(i)} - V_i(\vec{W}))^2 \quad .$$

De este modo, la "culpa" del error está compartida entre todos los pesos escogidos, y por lo tanto durante el proceso de ajuste, los pesos se deberán ir modificando según "cuánta culpa" tengan en el error.

Esto es, la información se propaga siempre hacia adelante en la red neuronal, generando una salida para determinadas entradas. Pero, durante el entrenamiento, además el error se propaga desde la salida hacia atrás, ajustando los pesos.

Aplicando método de gradiente, para cada peso individual, la fórmula de actualización será:

$$W_{(t+1)} = W_{(t)} + \alpha(\partial E / \partial W)|_{(Wt)} + \beta(W_{(t)} - W_{(t+1)})$$

Con α factor de aprendizaje y β factor de momentum, ambos entre 0 y 1.

Entonces, una vez creado el conjunto de entrenamiento y decidido un criterio de convergencia (cuándo debe terminar el entrenamiento), el proceso de entrenamiento se puede plantear en modo estructurado como sigue:

- Definir la arquitectura de la red (neuronas de entrada y salida, capas ocultas, neuronas por capa oculta).
- Definir los factores de aprendizaje y momentum.
- Inicializar la red con pesos aleatorios.
- Mientras no se alcance el criterio de convergencia:
  - Para i=1 hasta el número de instancias del conjunto de entrenamiento:
    - Pasar hacia adelante (feed-forward) las entradas de la i-ésima instancia.
    - Calcular el error en la i-ésima instancia.
    - Retropropagar el error y ajustar los pesos.
  - Continuar con el siguiente i.
  - Revisar si hay convergencia.
- Fin.

Normalmente, el conjunto de entrenamiento es dividido en, al menos, dos partes: un conjunto de entrenamiento, propiamente tal, y otro de validación.

De este modo, se obtiene un modelo construido con el conjunto de entrenamiento y se prueba el rendimiento del modelo construido, mediante el conjunto de validación (datos "desconocidos" para el modelo).

También el criterio de convergencia es relevante. Cuando los ciclos de

entrenamiento son muchos, existe el riesgo de tener una situación como la que se presenta en la Figura 7

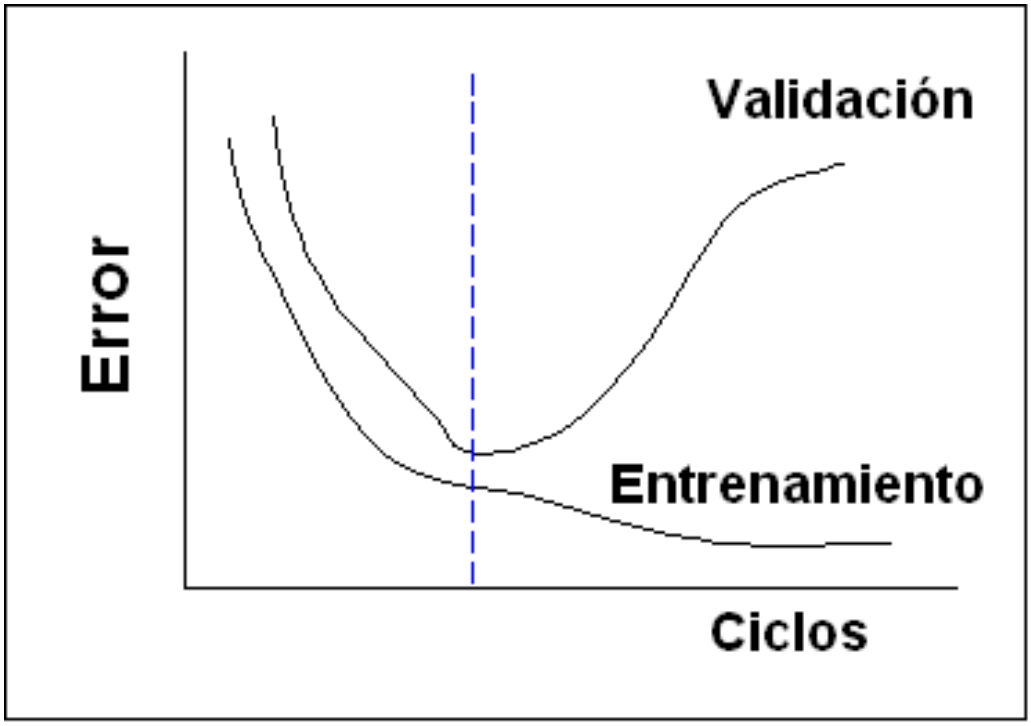


Figura 7: Curva de error en entrenamiento y validación en un caso con muchos ciclos de entrenamiento.

Se puede encontrar más información sobre el tema en [3] y en [7].

2.4.2 Máquinas de Soporte Vectorial.

La máquina de soporte vectorial (Support Vector Machine, SVM) es otro conocido algoritmo de clasificación.

Según se describe en [3], los clasificadores SVM están basados en los hiperplanos de la clase

$$(w \cdot x) + b = 0 \, , w \in \mathbb{R}^n \, , b \in \mathbb{R}$$

con la función de decisión $f(x) = sgn((w \cdot x) + b)$.

Entre todos los hiperplanos posibles, existe un único hiperplano óptimo que implica separación máxima entre las clases.

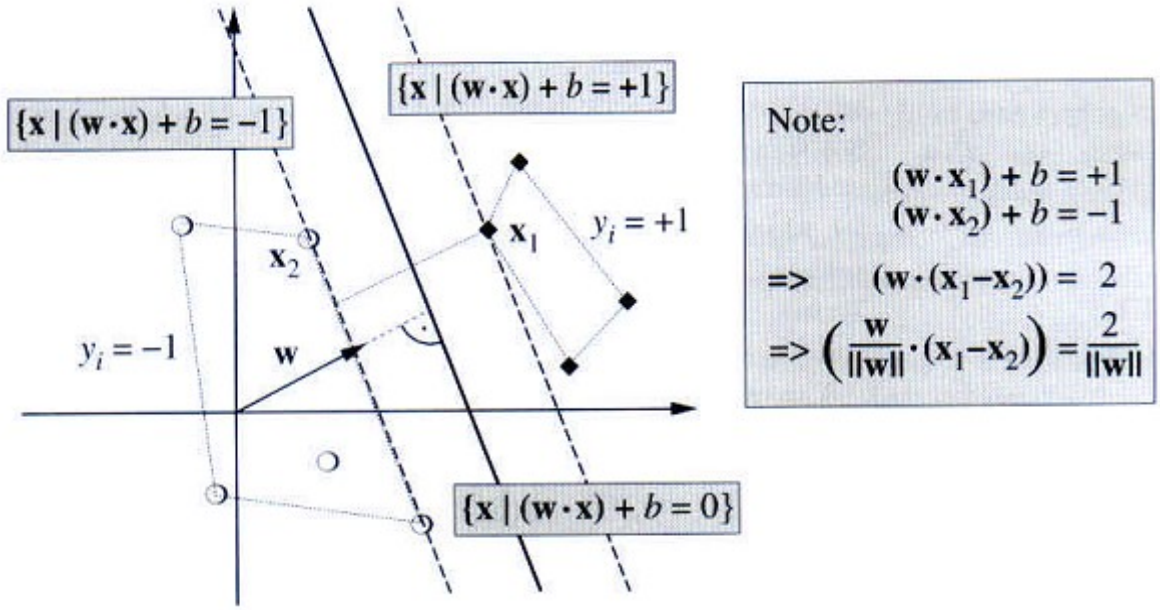


Figura 8: Problema de clasificación en dos dimensiones: separar círculos de rombos.

En la Figura 8 se aprecia un problema de clasificación en dos dimensiones (en tal caso, el hiperplano separador es una recta). El hiperplano separa a las dos clases en el espacio.

Si se considera el hiperplano óptimo y los dos hiperplanos paralelos a él que marcan los límites de ambas clases (a ambos lados del hiperplano óptimo, líneas punteadas en la Figura 8), si se traza la línea más corta que una a ambos hiperplanos límites, el hiperplano óptimo es ortogonal a esta y la intersecta justo a mitad de camino.

El problema de encontrar el hiperplano óptimo consiste en encontrar el hiperplano tal que la línea más corta que une a los hiperplanos límites tenga mayor longitud.

En la nota de la Figura 8 se puede apreciar que el márgen medido

perpendicularmente al hiperplano es $\frac{2}{\|w\|}$.

Para encontrar el hiperplano óptimo, se debe resolver el siguiente problema de optimización:

$$Min\tau(w)=\frac{\|w\|^2}{2} \quad ,$$

sujeto a: $y_i\cdot((w\cdot x)+b)\geq 1, i=1,\ldots,l\,,\ y_i\in\{\pm 1\}$.

Este problema se trata introduciendo multiplicadores de Lagrange (α_i), llegándose a :

$\sum \alpha_i y_i=0$ y $w=\sum \alpha_i y_i x_i$.

El vector solución tiene una expansión en términos de un subconjunto del conjunto de entrenamiento, aquellas instancias para las que α_i es distinto de cero. A estos patrones se les llama vectores de soporte.

Los vectores de soporte son los que están sobre el margen (ver Figura 9), y el resto de los patrones de entrenamiento resultan irrelevantes para determinar el hiperplano óptimo.

Si el problema no es linealmente separable, se utilizan funciones kernel para transformar el espacio y aplicar SVM, como se muestra en la Figura 9.

Existen varios tipos de funciones kernel. Algunos de los más utilizados son:

- Polinomial: $k(x,x')=\langle x,x'\rangle^d$
- Gaussiano: $k(x,x')=\mathrm{e}^{\frac{-(\|x-x'\|)}{(2\sigma^2)}}$
- Signoide: $k(x,x')=\tanh(\kappa\langle x,x'\rangle+\theta)$

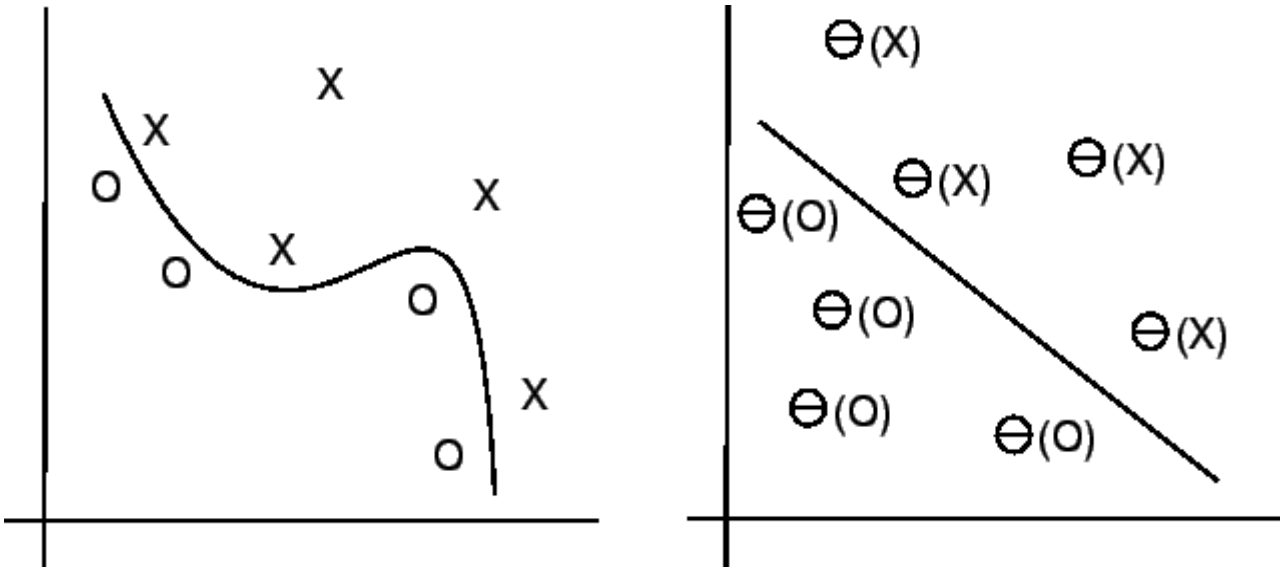


Figura 9: A la izquierda: el problema de clasificación no-lineal. No es posible trazar un hiperplano que separe las distintas clases de modo efectivo. A la derecha: el espacio transformado mediante una función de kernel, de modo que se pueda trazar un hiperplano sore el espacio transformado que separe de mejor forma las dos clases.

2.4.3 Tablas y Árboles de Decisión.

A diferencia de los algoritmos presentados anteriormente, hay otra clase que no son de "caja negra", sino que generan un set de reglas que se pueden conocer y, por ende, se puede saber qué va a decidir el algoritmo clasificador ante un determinado caso, y de acuerdo a qué parámetros tomará su decisión.

Los algoritmos de tabla de decisión generan reglas de tipo "*if...else...*" (*si... de lo contrario...*). Los árboles de decisión, en cambio, generan un árbol como el que se ejemplifica en la Figura 10.

Existen varios algoritmos de estos tipos. Se mencionan algunos a continuación.

RIDOR (RIpple-DOwn-Rules learner): Decide una clase por defecto como regla de nivel más alto. Luego decide excepciones para la regla por defecto, y sigue iterando hasta encontrar las mejores excepciones a las excepciones. Las excepciones son, finalmente, un set de reglas que predicen clases que no obedezcan a la clase por defecto.

Un ejemplo de clasificación con RIDOR se aprecia en la Figura 11.

C4.5: Un conocido algoritmo de árbol de decisión. Como se explica en [10] y en [1], este algoritmo genera un árbol y transforma el árbol en un set de reglas, individualizando cada camino desde la raiz hasta cada hoja como una regla individual. Posteriormente, prioriza las reglas (las somete a un ranking) para evitar conflictos. Finalmente, el algoritmo elimina reglas del set mientras esto haga disminuir el error en las instancias de entrenamiento.

Random Forest: Se explica en [10] que este algoritmo genera un "bosque". Es decir, un conjunto de árboles de decisión. Cada uno de los árboles decide un resultado de clasificación independiente de los otros. La salida final del algoritmo para cada instancia a clasificar será la moda de la decisión de todos los árboles del bosque.

RIPPER: Como se puede extraer de [1] y de [10], este es un algoritmo con estrategia de tipo "dividir para conquistar". Genera una regla a la vez, y aquellas instancias del set de entrenamiento que son cubiertas por estas reglas son eliminadas. Iterativamente deriva nuevas reglas para las instancias restantes. El resultado es un set de reglas de tipo *if..else*.

Una implementación de este algoritmo se aprecia en la Figura 12.

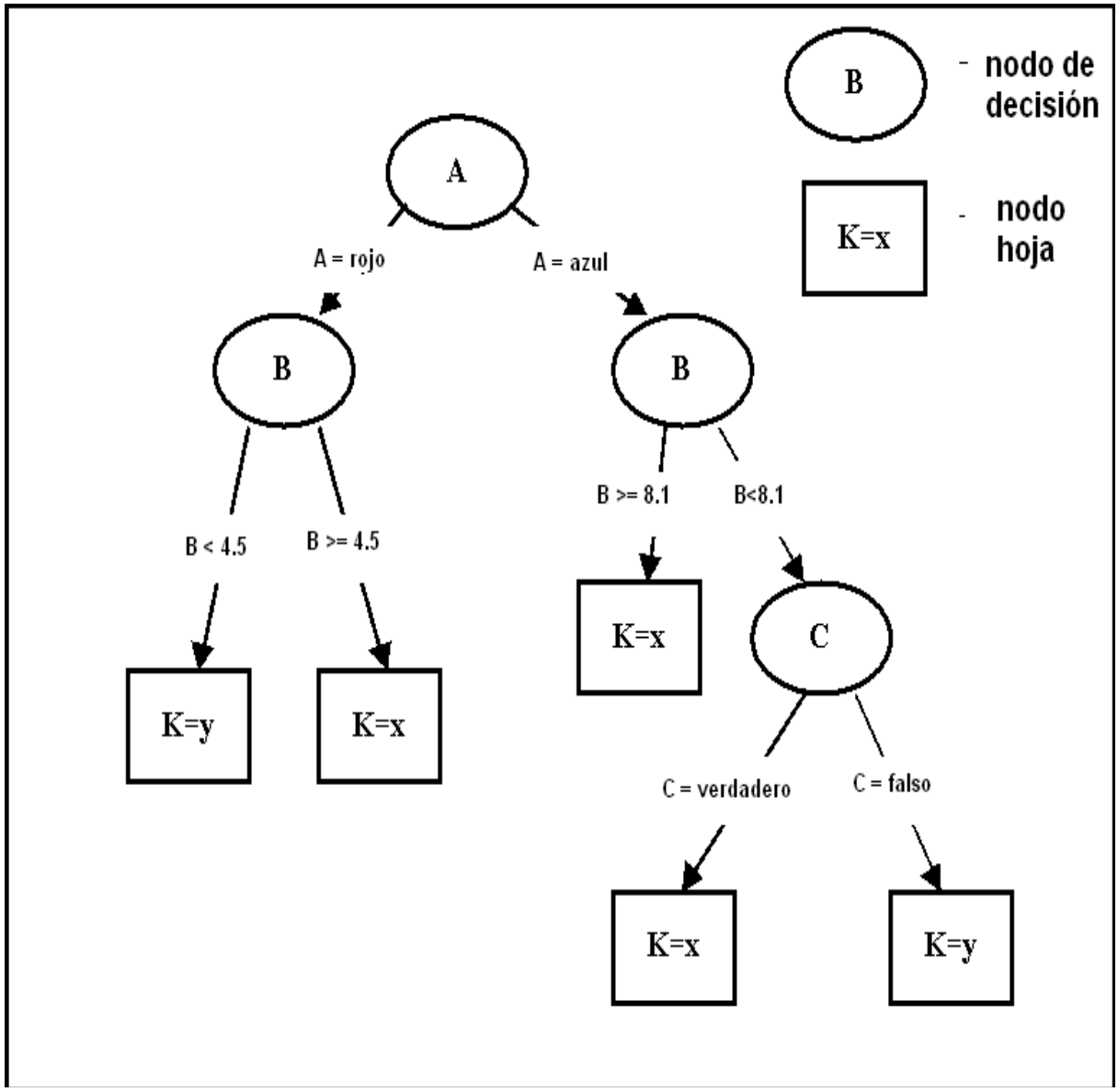


Figura 10: Árbol de decisión: determinar el valor de K, decidiendo segun los valores que tomen A,B y C.

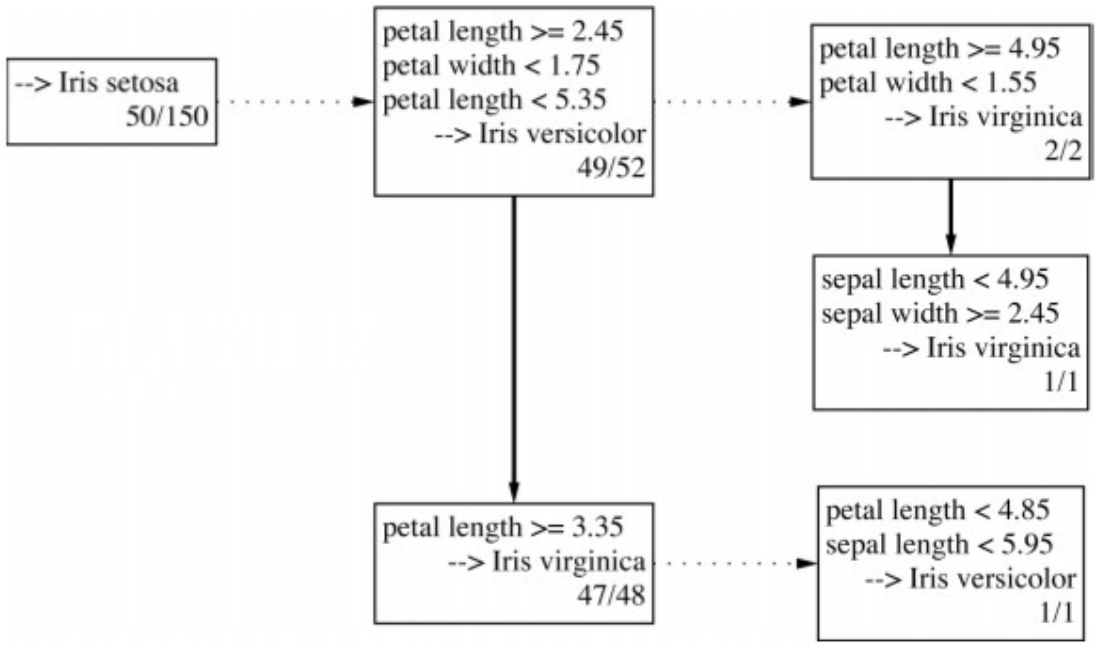


Figura 11: Ejemplo de clasificación de flores según características de pétalo y sépalo mediante RIDOR: Regla por defecto y sucesivas excepciones.

PART: Algoritmo de árboles de decisión parciales. Genera, un set de reglas de acuerdo a la estrategia de dividir para conquistar. Elimina todas las instancias del set de entrenamiento cubiertas por estas reglas, y procede recursivamente hasta que no queden instancias. Para generar cada regla, PART construye un "árbol de decisión parcial" para el set de instancias actual (descartando aquellas que hayan sido cubiertas por reglas anteriormente), y escoge la hoja con la mayor cobertura como nueva regla. Luego de esto, el árbol parcial es descartado (como se explica en [1]). Al igual que en el caso de RIPPER, el resultado de la aplicación de este algoritmo en el entrenamiento es un set de reglas de tipo *if...else,* según las cuales se evaluará cada instancia a clasificar.

Un ejemplo de set de reglas, como los entregados por PART, se muestra en la Figura 13.

```
procedure BuildRuleSet(P,N)
P = positive examples
N = negative examples
RuleSet = {}
DL = DescriptionLength(RuleSet, P, N)
while P ≠ {}
      // Grow and prune a new rule
      split (P, N) into (GrowPos, GrowNeg) and (PrunePos, PruneNeg)
      Rule := GrowRule(GrowPos, GrowNeg)
      Rule := PruneRule(Rule, PrunePos, PruneNeg)
      add Rule to RuleSet
      if DescriptionLength(RuleSet, P, N) > DL + 64 then
            // Prune the whole rule set and exit
            for each rule R in RuleSet (considered in reverse order)
                  if DescriptionLength(RuleSet − {R}, P, N) < DL then
                        delete R from RuleSet
                        DL := DescriptionLength(RuleSet, P, N)
                        end if
                  end for
            return (RuleSet)
            end if
      DL := DescriptionLength(RuleSet, P, N)
      delete from P and N all examples covered by Rule
      end while
end BuildRuleSet

procedure OptimizeRuleSet(RuleSet, P, N)
for each rule R in RuleSet
      delete R from RuleSet
      UPos := examples in P not covered by RuleSet
      UNeg := examples in N not covered by RuleSet
      split (UPos, UNeg) into (GrowPos, GrowNeg) and (PrunePos, PruneNeg)
      RepRule := GrowRule(GrowPos, GrowNeg)
      RepRule := PruneRule(RepRule, PrunePos, PruneNeg)
      RevRule := GrowRule(GrowPos, GrowNeg, R)
      RevRule := PruneRule(RevRule, PrunePos, PruneNeg)
      choose better of RepRule and RevRule and add to RuleSet
      end for
end OptimizeRuleSet

procedure Ripper(P, N, k)
RuleSet := BuildRuleSet(P, N)
repeat k times RuleSet := OptimizeRuleSet(RuleSet, P, N)
return (RuleSet)
end Ripper
```

Figura 12: Presentación en pseudocódigo de algoritmo RIPPER.

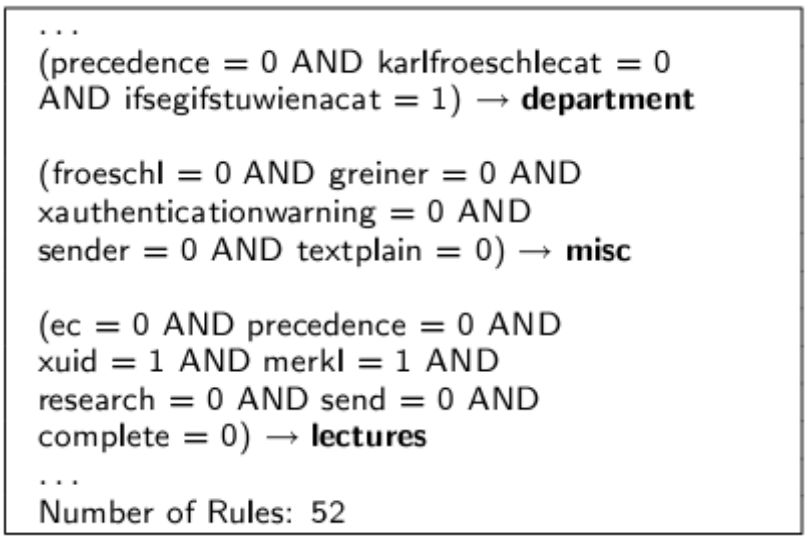


Figura 13: Ejemplo de reglas de decisión presentado por Berger, Merkl y Dittenbach.

Varios de estos algoritmos fueron probados en la realización de esta memoria. Los mejores resultados fueron obtenidos con PART, por lo que, en adelante, se presentarán de modo comparativo los resultados obtenidos con redes neuronales, SVM y PART por el lado de las tablas de decisión.

3 Diseño de la solución.

3.1 Diseño propuesto

A diferencia del algoritmo antiguo, el nuevo diseño evita la comparación directa de strings. Por el contrario, pretende modelar el problema de comparación de strings de caracteres como un problema numérico.

La filosofía tras la solución planteada consiste en traducir cada par de strings a comparar en una serie de valores numéricos que den cuenta de su posible similitud. Estos valores numéricos serán llamados ***características***. En adelante, a cada conjunto de valores numércos calculados para cada par de strings a comparar, se llamará ***set de características***.

Posteriormente, cada set de características es pasado por un algoritmo clasificador el cual, según los valores de estas características, decidirá a qué clase pertenece cada entrada (similitud o no similitud).

El problema puede ser descrito en modo estructurado:

1. *Obtener la lista de nuevas solicitudes registradas en NIC Chile y guardarlas en una tabla.*
2. *Se toma un String protegido*
3. *Se toma un String de la tabla de nuevas solicitudes.*
4. *Para el par de Strings protegido/nueva solicitud que se ha tomado, se calcula un set de características.*
5. *Si las características no son distintas de nulo (es decir, si es que hay alguna posibilidad de similitud), se guarda el set de características en la tabla correspondiente.*
6. *Si quedan strings en la tabla de nuevas solicitudes, se vuelve al punto 3, con el siguiente string de la tabla.*
7. *Si quedan strings en la lista de protegidos, se vuelve al punto 2, con el*

siguiente de la lista.

8. *Se toma un set de características de la tabla de características a clasificar.*
9. *Se aplica algoritmo clasificador.*
10. *Si el algoritmo determina que pertenece a la clase "similitud", guarda la referencia en la tabla de similitudes a reportar.*
11. *Si quedan más entradas en la tabla de características a clasificar, se vuelve al punto 8.*
12. *Finaliza.*

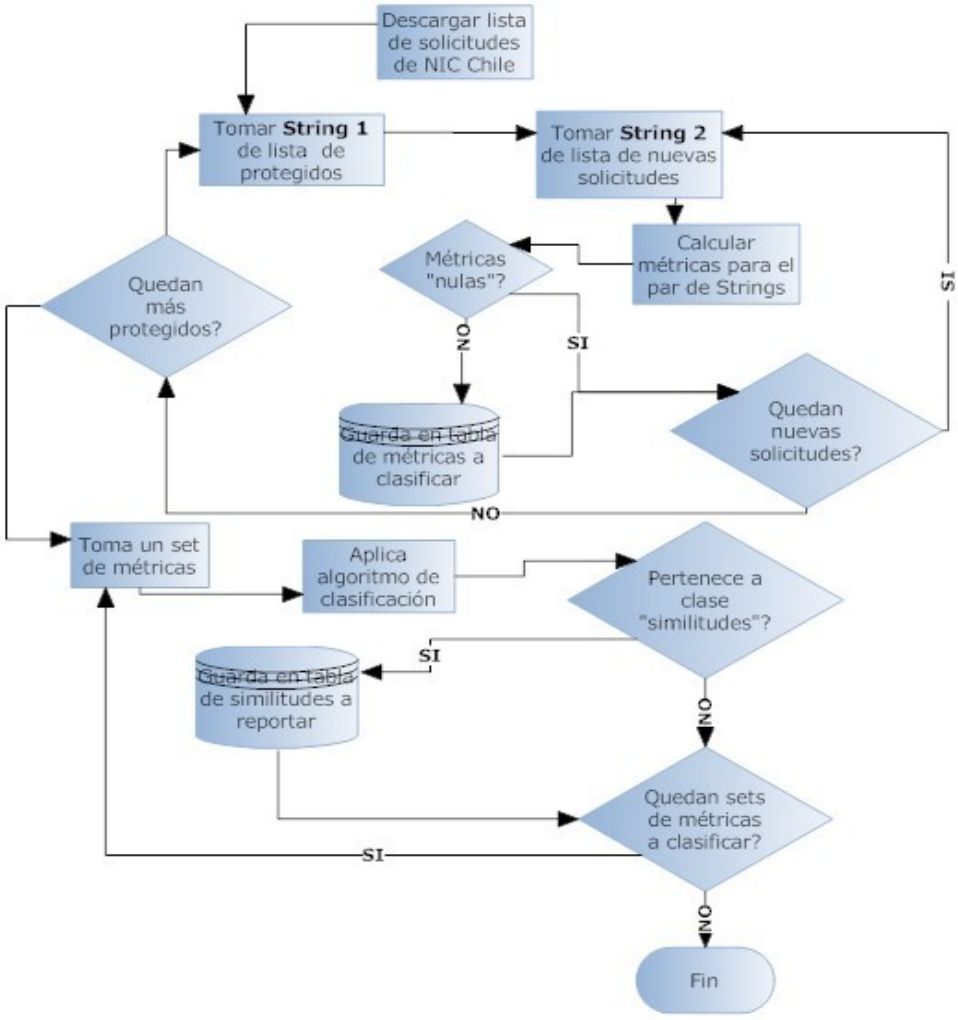


Figura 14: Diagrama de flujo del algoritmo nuevo.

También se puede representar en la forma de diagrama de flujo, como se muestra en la Figura 14.

En secciones posteriores de esta memoria se presentarán los resultados obtenidos mediante este diseño, y la comparación en contraste con el algoritmo previo.

Será necesario presentar cuantitativamente estos resultados y comparaciones, y hacer un análisis riguroso de ellas, además de cuantificar, para ambos algoritmos, los errores de cada tipo: similitudes que se escapan y alertas que no constituyen similitudes reales.

3.2 Comparación y cálculo de características.

Para comparar dos Strings y calcular las características que permitirán su clasificación, el algoritmo diseñado opera como se explicará a continuación.

Sea *String 1* una cadena de caracteres correspondiente a un concepto en vigilancia (protegido) y sea *String 2* una cadena de caracteres correspondiente a una nueva solicitud de dominio de internet.

Se considera una similitud cuando String 1 está contenido (exacta o aproximadamente) en String 2. Es decir, una similitud es tal cuando el String 1 es igual o parecido a String 2 o cuando String 1 es subconjunto exacto o aproximado de String 2.

Por ejemplo:

Similitud caso 1:

- String 1: <marca>.
- String 2: <marca>.

$$String\,1 = String\,2$$

Similitud caso 2:

- String 1: <marca>.
- String 2: <marca>chile.

$$String\,1 \subset String\,2$$

De este modo, se buscará inicialmente si String 1 coincide (hace matching) aproximadamente con un subconjunto de String 2 (que puede ser el String 2 completo). Luego, se calcularán las características que permitan calcular si String 1 y el subconjunto de String 2 efectivamente constituyen una similitud.

El paso inicial es importante, porque permite identificar, dentro de String 2, las componentes. Sea, de ahora en adelante, *subString2m* el substring de String 2 con el cual String 1 hace matching. Sean *subString2a* y *subString2b* las cadenas de caracteres externas, tales que String2 está compuesto de la siguiente forma:

subString2a+subString2m+subString2b.

Por ejemplo:

- String 1: <marca>
- String 2: la<marca>chile
- subString2a: la
- subString2b: chile
- subString2m: <marca>

3.3 Características.

Las características a utilizar se definieron mediante el siguiente procedimiento:

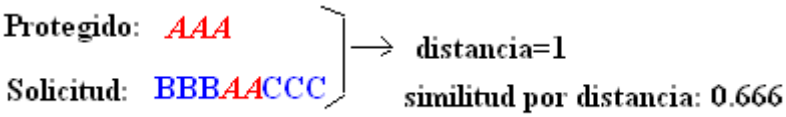
- Observación de cualidades comunes de grandes volúmenes de pares de strings, tanto similares como no similares. La mayoría de los casos observados fueron extraídos de la base de datos del sistema de vigilancia antiguo.
- Hipótesis sobre la base de las observaciones realizadas. Por ejemplo: "la distancia de edición es un factor que permite discriminar entre similitudes y no-similitudes".
- Experimentación para probar o descartar la hipótesis. Se aplica a grandes volúmenes de casos (ya sea manualmente o mediante alguna herramienta de software programada especialmente para dicho fin). Por ejemplo: Se programa una herramienta de software que calcula la distancia de edición entre un par de strings y se aplica a varios de los casos observados. Se observan los resultados del experimento.
- Se acepta o rechaza la hipótesis según la experimentación.

Del modo anteriormente descrito, se llega a definir el set de características que se explica a continuación.

3.3.1 Distancia de edición parcial.

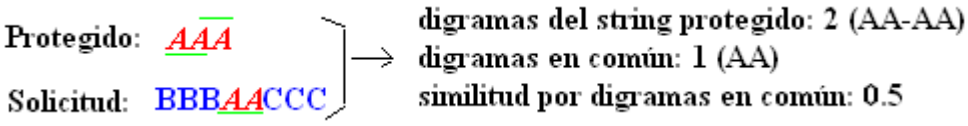
Similitud basada en la distancia de edición entre String 1 y subconjunto subString2m de String 2:

Se calcula la distancia de edición entre String 1 y subString2m y se obtiene un valor normalizado del grado de similitud basado en esta distancia.



3.3.2 Digramas comunes.

Se cuentan los digramas en común entre String 1 y subString2m. Se calcula, tal como para el caso anterior, un grado de similitud normalizado para este parámetro.



3.3.3 Distancia de edición completa.

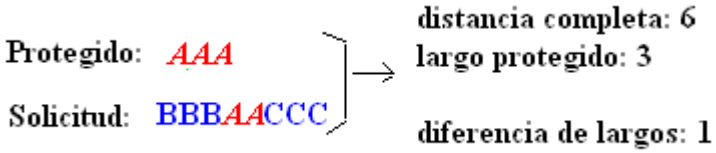
Del mismo modo que la distancia de edición parcial, pero comparando String 1 con String 2 directamente, no con subString2m. El valor se normaliza en el rango [0,1].

3.3.4 Largo de String protegido.

La observación reveló que la cantidad de caracteres que componen String 1 suma o resta puntos porcentuales a la probabilidad de que un par de strings constituya una similitud.

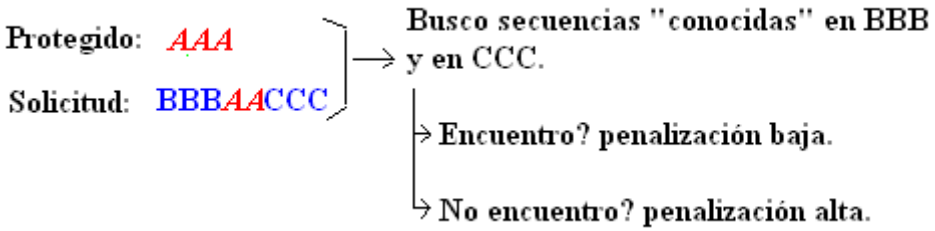
3.3.5 Diferencia de largos.

La diferencia de cantidad de caracteres entre String 1 y subString2m también es un factor que aporta a la detección de similitudes, particularmente para controlar casos de inserciones de caracteres en medio.



3.3.6 Penalización.

Se calcula una penalización para los substrings subString2a y subString2b según correspondan a acompañantes atractivos o no atractivos.



La observación de casos reveló que un gran número de las similitudes reportadas por el sistema antiguo, corresponden a casos en que String 1 es parte de una palabra contenida en String 2. Por ejemplo:

Caso 1.

String 1: *mar*.

String 2: *mar*tillo.

En este caso subString2m sería *mar*, lo cual constituye un matching exacto con String 1. Sin embargo, se desea que el sistema **no reporte** este caso como similitud.

Caso 2.

String 1: *mar*.

String 2: el*mar*.

En este segundo caso, existe un matching igual que en el caso anterior. Este caso es muy probable que si se quiera reportar como similitud.

La diferencia está en las secuencias de caracteres externas. En particular, subString2a constituye un acompañante atractivo, que induce a una mayor probabilidad de generar una similitud, y por lo tanto el algoritmo le asignará una penalización menor que las asignadas a la secuencia externa (subString2b) del caso 1.

La solución adoptada consiste en almacenar una tabla denominada "diccionario", en la cual se almacenan acompañantes típicos, basados en la observación de casos históricos.

En esta tabla se incluyen casos como "el", "la", "chile", "limitada", "banco", "radio", etcétera.

Las secuencias de caracteres externas se contrastan contra esta lista de términos. A los caracteres que coincidan con términos del diccionario, se les asigna una penalización baja.

En cambio, a los términos que no coincidan con términos almacenados en el diccionario, se les asignará una penalización mayor, bajándole de este modo al par de strings la probabilidad de que sea considerado como similitud.

El criterio de penalización es el siguiente:

- Secuencias de caracteres incluídas en el diccionario: penalización = 1.
- Secuencias de caracteres no incluidas en el diccionario: penalización = $5 \cdot l$, en que l es el largo en caracteres de la secuencia.

4 Metodología

4.1 Herramientas utilizadas.

Dada la naturaleza del trabajo, se deben mencionar las herramientas de software utilizadas en su desarrollo.

Las herramientas utilizadas fueron, básicamente, dos: el lenguaje de programación y la herramienta de prueba de algoritmos de aprendizaje. Todas las otras herramientas secundarias, utilizadas durante el diseño y para hacer pruebas, fueron programadas especialmente para cada requerimiento.

El lenguaje de programación escogido para la implementación fue java. Los motivos para escoger este lenguaje fueron varios, entre los cuales se puede contar:

- Orientación a objetos.
- Posibilidad de portar la solución a plataformas distintas (MS Windows, Linux, etcétera).
- Razonable comodidad para trabajar con strings y con bases de datos,
- Extensión de sus posibilidades mediante la instalación de librerías,
- Capacidad de manejar todas las operaciones contempladas en el diseño inicial de la solución.

Para el trabajo con algoritmos de aprendizaje, se escogió la aplicación *Weka*, desarrollada en la universidad de Waikato (Nueva Zelanda). Los motivos fueron, principalmente:

- Recurrentes menciones en papers y publicaciones en que se comparaban algoritmos de clasificación.
- Amplia variedad de algoritmos de clasificación.
- Implementación de librerías en java (mismo lenguaje de programación utilizado para desarrollar el trabajo), lo cual favorece la integración con

el resto del trabajo.

4.2 Construcción de los conjuntos de entrenamiento y pruebas.

La principal ventaja de los algoritmos de clasificación probados (y del que se implementó finalmente) es su capacidad de aprender mediante el sometimiento a un proceso de entrenamiento.

Para entrenar a estos algoritmos, es necesario construir antes un conjunto de entrenamiento lo suficientemente representativo del universo a clasificar y que aporte suficiente información para lograr generar un modelo confiable para la clasificación de casos desconocidos, no incluidos en este conjunto de entrenamiento.

El proceso consta de tres fases, en cada una de las cuales se utiliza un conjunto distinto:

- Entrenamiento, para ajustar los parámetros de los modelos (o generar las reglas, en caso de los algoritmos de tipo tablas de decisión).
- Validación, para obtener los porcentajes de éxito de los modelos, matrices de confusión, etcétera, y decidir qué modelos son los más apropiados.
- Pruebas independientes (en adelante referidas simplemente como "pruebas"), para estudiar los resultados "reales" entregados por el modelo escogido, sin modificar ya el modelo, y realizar las comparaciones de rigor con el algoritmo antiguo.

Los sets de datos de las fases de entrenamiento y validación se construyeron juntos, como un solo conjunto, dividiendo *a posteriori*. El conjunto para la fase de pruebas no fue directamente construido: para esta fase simplemente se obtuvieron los datos reales (pares de strings dominio solicitado / string protegido) obtenidos diariamente durante un período de

tiempo definido.

La construcción del conjunto de entrenamiento y validación se basó en dos pilares fundamentales:

- *Observación de casos históricos registrados en la base de datos del sistema.* Se observan casos pasados que en su momento fueron marcados como similitudes y como no-similitudes. Se incluyen casos, intentando escoger casos que sean parecidos a muchos otros, de modo de incluir muestras que resulten representativas. Además de estos casos reales registrados, se incluyen algunas variaciones sutiles sobre estos casos, otorgando a cada una de las instancias incluidas la clasificación de similitud o no-similitud. En los casos históricos reales, la aplicación de esta clasificación es simplemente extraída del resultado almacenado en la base de datos. En cuanto a las variaciones, se opera, según la complejidad del caso, asignando la clase según "sentido común" o, en otros casos menos simples, consultando el criterio que aplicarían los operadores del sistema.
- *Conversación con los operadores del sistema.* Dado que los operadores del sistema, aquellas personas que han tenido contacto con él todos los días durante los últimos, al menos, cuatro años son quienes mejor conocen el sistema, los casos posibles que se puedan registrar y los criterios según los cuales estos se clasificarían, se intentó aprovechar esta experiencia, preguntándoles directamente por casos comunes y extraños que se pudiesen clasificar como similitudes o no-similitudes, así como por casos relevantes reportados erróneamente por el sistema antiguo (en particular, omisiones relevantes). Además de las consultas directas, se observó el accionar diario de los operadores del sistema, así como sus diálogos y discusiones al respecto. Los datos relevantes extraídos de estas consultas y observaciones fueron utilizados en la construcción del conjunto de entrenamiento.

Luego de que este proceso fue completado y ya se realizaron los primeros intentos de validación en un modo preliminar, se incluyeron además algunos casos mal clasificados en estos primeros intentos al conjunto de entrenamiento, y luego se volvieron a generar los modelos de modo definitivo.

Mediante este procedimiento se generó un conjunto total (entrenamiento más validación) de 3858 instancias, el cual, a su vez, fue partido arbitrariamente en dos, generándose así el conjunto de entrenamiento y un conjunto de validación.

Del conjunto total, se debe señalar que 1968 instancias corresponden a no similitudes, mientras 1890 corresponden a similitudes. Es decir, se encuentra levemente desbalanceado, siendo el 52% no similitudes y el 48% similitudes.

Una vez realizadas las fases de entrenamiento y validación, se realizaron pruebas con datos nuevos. Los datos nuevos utilizados en las pruebas fueron, como se explicó previamente, datos reales obtenidos diariamente de las nuevas solicitudes de dominios de internet registradas en la base de datos de NIC Chile y de los dominios protegidos por la empresa.

5 Resultados.

5.1 Relevancia de cada característica en la clasificación.

Antes de presentar los resultados de validación y pruebas, se intentará mostrar mediante gráficos la relevancia individual de cada una de las características utilizadas en la clasificación, para ver si es que cada una de ellas es significativa en la solución del problema, o si alguna aporta solamente ruido.

Este estudio se realiza sobre un universo de más de 3000 casos, graficando, para cada característica, el valor numérico de la característica contra el número de muestras clasificadas como "similitud" o "no similitud".

Para cada uno de los gráficos presentados, los resultados graficados en rojo corresponden a similitudes, mientras los mostrados en azul corresponden a no-similitudes.

5.1.1 Similitud por digramas comunes.

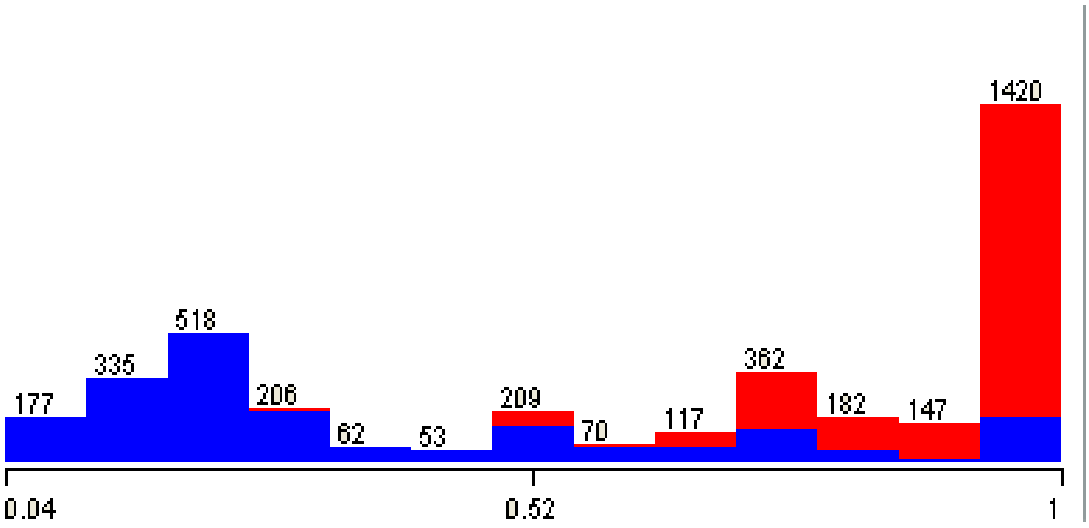


Figura 15: Similitud por digramas comunes.

En la Figura 15 se puede apreciar que la similitud por digramas, de modo individual, incide claramente en la clasificación de casos. Se aprecia en el

gráfico que, para similitud por digramas cercana a cero (es decir, "menos digramas en común"), los casos correspondientes a no-similitudes son muy mayoritarios. En cambio, al acercarse el valor a uno (es decir, "más digramas en común") los casos correspondientes a similitud son claramente mayoritarios.

Es en el rango superior a 0.7 que los casos clasificados como similitud predominan notoriamente sobre aquellos clasificados como no-similitud.

5.1.2 Similitud por distancia de edición parcial.

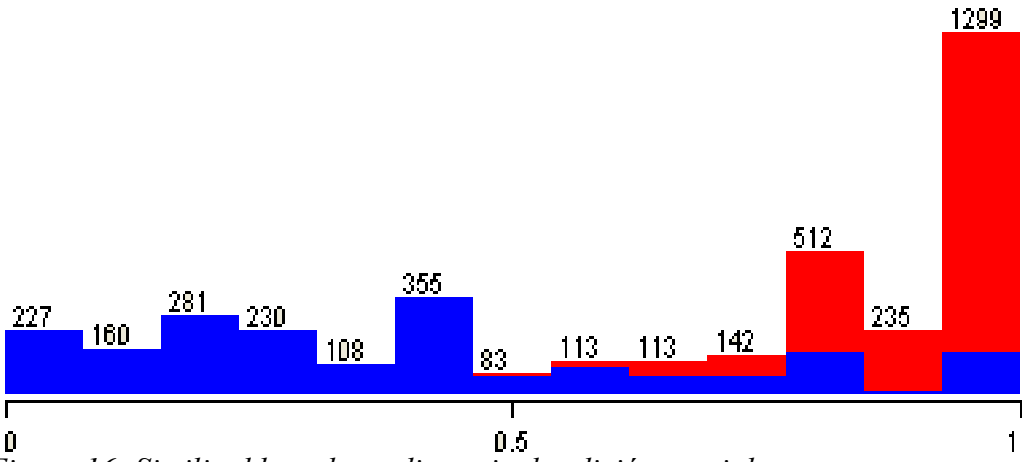


Figura 16: Similitud basada en distancia de edición parcial.

Se aprecia en la Figura 16 que la similitud basada en distancia de edición parcial es claramente relevante para separar casos entre similitudes y no similitudes. El gráfico permite apreciar que, para valores menores que 0.5 de la característica, los casos son casi en su totalidad no-similitudes.

En cambio, para casos con la característica cercana a 1 (con exactitud, para el rango sobre 0.769 en adelante), son en su gran mayoría similitudes.

5.1.3 Distancia de edición total.

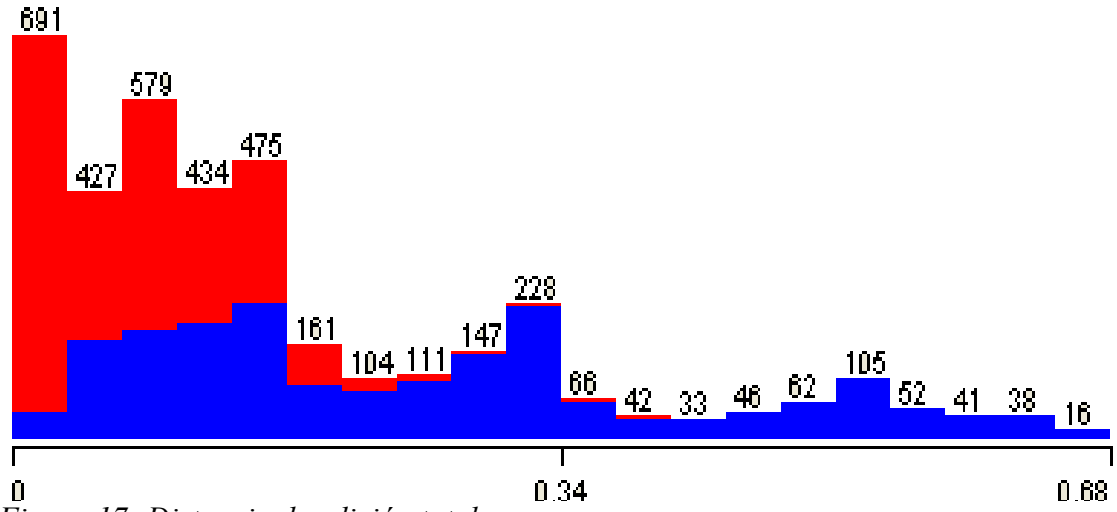


Figura 17: Distancia de edición total.

Se aprecia en la Figura 17 que la distancia de edición total también difiere notablemente para casos clasificados como similitud o no-similitud.

En efecto, para valores superiores a 0.34 de la distancia de edición total (normalizada) los casos son, casi en su totalidad, no-similitudes, mientras para valores cercanos a cero, los casos son mayoritariamente similitudes.

5.1.4 Diferencia de largos.

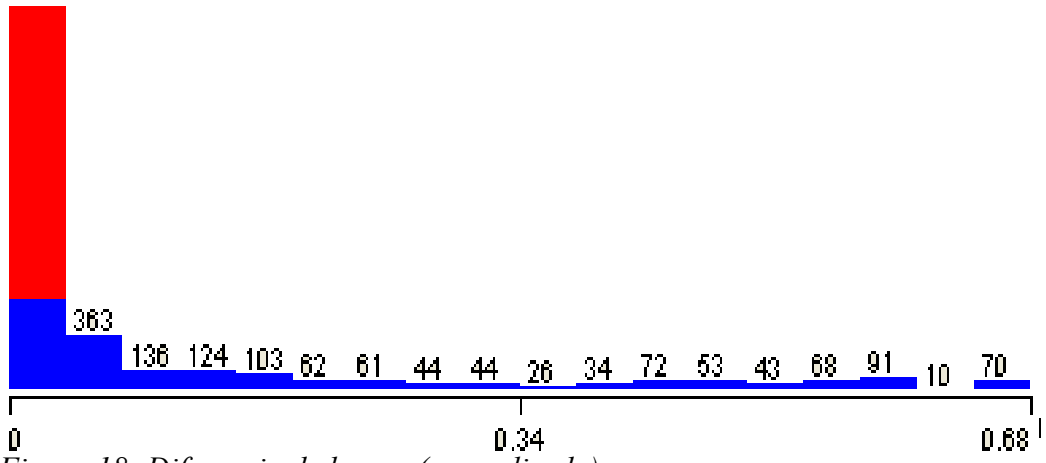


Figura 18: Diferencia de largos (normalizada).

Se aprecia en la Figura 18 que la mayoría de casos correspondientes a similitudes se concentran en rangos de valores pequeños para el valor de la diferencia de largos entre String 1 y subString2m. El primer rango que aparece en la gráfica, en que se concentran todos los casos correspondientes a similitudes, es el rango de valor 0 hasta 0.038 de la característica. Fuera de este rango, la totalidad de los valores corresponden a no-similitudes.

5.1.5 Largo del string protegido.

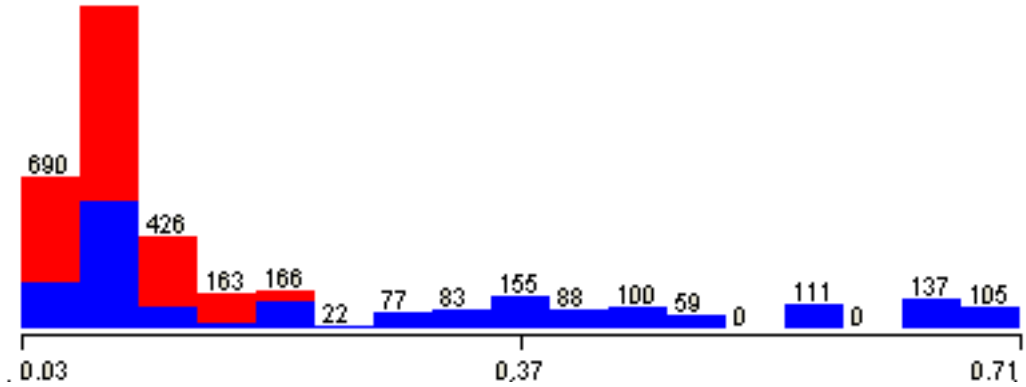


Figura 19: Largo del string protegido, String1.

En el caso de esta característica, se puede apreciar que, en general, los casos correspondientes a similitudes se concentran en valores pequeños (inferiores a 0.37). Sin embargo, lo realmente interesante será ver cómo opera esta característica en conjunto con las otras.

5.1.6 Penalización.

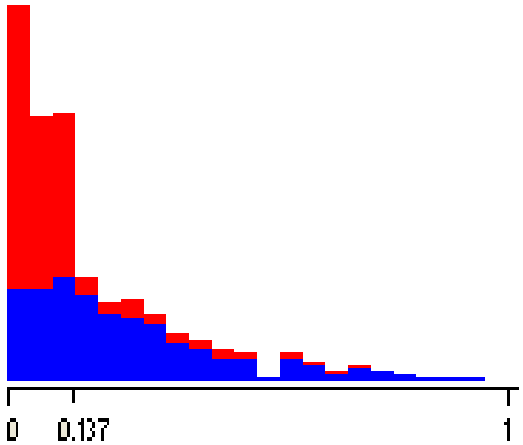


Figura 20: Penalización (normalizada).

Se puede ver en la gráfica que la mayoría de los casos correspondientes a similitudes se concentran en valores pequeños (inferiores a 0.137) de la característica. Para valores que superan este umbral, la mayoría de los casos son clasificados como no-similitud.

5.1.7 Resumen: relevancia de las características de forma individual.

Con lo anterior, se puede apreciar que cada una de las características escogidas, de modo individual, significa un aporte a la clasificación. Sin embargo, lo realmente relevante, es el aporte que puedan significar de modo conjunto y cómo se deben combinar para poder efectivamente discriminar casos.

En el marco del presente problema, esta labor de combinación de las seis características utilizadas no es realizada por un humano, sino por los distintos algoritmos de clasificación probados, en los respectivos procesos de entrenamiento.

5.2 Generación de modelos.

Se probó generar varios modelos, de los tipos mencionados previamente en este documento. A continuación, se muestran los resultados de los modelos obtenidos, con el mismo conjunto de entrenamiento, para los distintos tipos de modelos. Se construyó cada tipo de modelo 10 veces, obteniendo cada vez (en el proceso de validación) el porcentaje de éxito y el error estimado del modelo, además de la matriz de confusión correspondiente. Se intentará comparar los resultados obtenidos con cada modelo, señalando qué modelo promete mejores resultados.

5.2.1 MLP

Se entrenó diez veces el algoritmo para obtener un clasificador de tipo red neuronal perceptrón multicapa, siempre utilizando el mismo conjunto de entrenamiento. Los resultados del entrenamiento y validación fueron los siguientes:

5.2.1.1 Primera ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	584	94.4984 %
Instancias mal clasificadas	34	5.5016 %

Tabla 1: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                    ------------
                    293  23 |  no
                     11 291 |  si
```

Figura 21: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 293 verdaderos negativos y 291 verdaderos positivos. Estas son las instancias bien clasificadas y suman 584.

Por otro lado, los falsos negativos son 11 y los falsos positivos ascienden a 23. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 34.

De este modo, se aprecia que, en esta ejecución, el porcentaje de instancias bien clasificadas es alto, cercano al 95%. El otro dato importante es el detalle de las instancias mal clasificadas. Los falsos positivos son más que los falsos negativos (más del doble, de hecho). Para este problema en particular, garantizar que los falsos negativos estén en baja proporción es importante: se debe recordar que un falso negativo corresponde a una similitud real que fue descartada, lo cual es altamente indeseable. Por otra parte, un falso positivo corresponde a una instancia que no constituye similitud, pero que fue reportada como similitud, un tipo de error que se pretende minimizar, pero que es menos crítico que el otro (para este problema en particular).

5.2.1.2 Segunda ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	576	93.2039 %
Instancias mal clasificadas	42	6.7961 %

Tabla 2: Porcentajes de éxito y de error.

Clasificadas como:	no	si	
	277	36	no
	6	299	si

Figura 22: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 277 verdaderos negativos y 299 verdaderos positivos. Estas son las instancias bien clasificadas y suman 576.

Por otro lado, los falsos negativos son 6 y los falsos positivos ascienden a 36. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 42.

En esta ejecución, el porcentaje de éxito que se predice para el modelo es levemente menor al obtenido en la ejecución anterior (una diferencia inferior al 2%).

Sin embargo, la matriz de confusión entrega información relevante sobre los resultados que se pueden esperar del modelo generado en esta ejecución: en efecto, el porcentaje de error es mayor, pero el aumento se aprecia por el lado de los falsos positivos. En cambio, por el lado de los falsos negativos el porcentaje disminuye.

Esto quiere decir que el modelo generado en esta segunda ejecución, si bien entregaría un mayor número de similitudes irrelevantes, omitiría un menor número de similitudes relevantes que el modelo generado en la primera ejecución.

5.2.1.3 Tercera ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	574	92.8803 %
Instancias mal clasificadas	44	7.1197 %

Tabla 3: Porcentajes de éxito y de error.

Clasificadas como:	no	si	
	316	28	no
	16	258	si

Figura 23: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 316 verdaderos negativos y 258 verdaderos positivos. Estas

son las instancias bien clasificadas y suman 574.

Por otro lado, los falsos negativos son 16 y los falsos positivos ascienden a 28. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 42.

En esta ejecución, el porcentaje de éxito que se predice para el modelo es menor al obtenido en las dos ejecuciones anteriores.

La matriz de confusión revela que, con respecto a los modelos anteriores, el porcentaje de error aumenta, entregando un mayor número de falsos negativos. Es decir, omite un mayor número de similitudes relevantes en la clasificación. Los falsos positivos, si bien son más que en la primera ejecución, son menos que en la segunda. Es decir, se espera que este modelo entregue menos similitudes irrelevantes que el generado en la segunda ejecución, si bien la cantidad de similitudes relevantes omitidas aumenta, lo cual no es deseable.

5.2.1.4 Cuarta ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	577	93.3657 %
Instancias mal clasificadas	41	6.6343 %

Tabla 4: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                   ---------+---
                    288  34 | no
                      7 289 | si
```

Figura 24: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 288 verdaderos negativos y 289 verdaderos positivos. Estas son las instancias bien clasificadas y suman 577.

Por otro lado, los falsos negativos son 7 y los falsos positivos ascienden a 34. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 41.

En esta ejecución, el porcentaje de éxito que se predice para el modelo es superior los dos anteriores (aún inferior al primero).

La matriz de confusión revela una comparación favorable en relación a los modelos obtenidos en las ejecuciones anteriores: el porcentaje de errores de ambos tipos obtenidos es, comparativamente, bajo. Solamente la primera ejecución ofrece un mejor resultado en este sentido.

Por otra parte, con respecto a la primera ejecución, este modelo entregaría un mayor número de similitudes irrelevantes, pero la cantidad de similitudes relevantes omitidas (falsos negativos) es uno de los más bajos obtenidos hasta el momento.

5.2.1.5 Quinta ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	585	94.6602 %
Instancias mal clasificadas	33	5.3398 %

Tabla 5: Porcentajes de éxito y de error.

Clasificadas como:	no	si	
	289	22	no
	11	296	si

Figura 25: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 289 verdaderos negativos y 296 verdaderos positivos. Estas son las instancias bien clasificadas y suman 585.

Por otro lado, los falsos negativos son 11 y los falsos positivos ascienden a 22. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 33.

Esta ejecución entrega el modelo con la mejor expectativa de éxito hasta el momento, con un porcentaje muy similar al obtenido en la primera ejecución.

La matriz de confusión revela más similitudes con la primera ejecución: la cantidad de falsos negativos es la misma, lo que quiere decir que, en la validación, el modelo omitió el mismo número de similitudes relevantes.

El número de falsos positivos es levemente inferior al obtenido en la primera ejecución. El modelo, entonces, reportaría como relevantes un número similar de similitudes irrelevantes. Es el mejor resultado en cuanto a falsos positivos de los registrados hasta el momento.

5.2.1.6 Sexta ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	588	95.1456 %
Instancias mal clasificadas	30	4.8544 %

Tabla 6: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                   ---------+---
                    312  14 | no
                     16 276 | si
```

Figura 26: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 312 verdaderos negativos y 276 verdaderos positivos. Estas son las instancias bien clasificadas y suman 588.

Por otro lado, los falsos negativos son 16 y los falsos positivos ascienden a 14. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman

30.

El modelo generado en esta ejecución supera a todos los anteriores en cuanto a porcentaje de éxito.

La matriz de confusión revela, sin embargo, que la cantidad de falsos negativos entregada por el modelo es una de las más altas entre todos los modelos generados hasta el momento. Esto es altamente indeseable.

El número de falsos positivos, en cambio, es el más bajo de todos los obtenidos hasta el momento. Esto quiere decir que el modelo generado es el que más reduce la cantidad de alarmas irrelevantes obtenidas. Sin embargo, la cantidad de similitudes relevantes omitidas es más del doble del entregado por otros de los modelos generados, lo cual resulta problemático. El modelo entrega el menor porcentaje de error, pero es un modelo más bien "estricto": tiende a descartar casos como no-similitudes, lo sean o no, aumentando el porcentaje de verdaderos positivos, pero aumentando también los falsos negativos.

5.2.1.7 Séptima ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	568	91.9094 %
Instancias mal clasificadas	50	8.0906 %

Tabla 7: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                    ------------
                    309  39 | no
                     11 259 | si
```

Figura 27: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a “no-similitud”.

Si corresponde a “similitud”.

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 309 verdaderos negativos y 259 verdaderos positivos. Estas son las instancias bien clasificadas y suman 568.

Por otro lado, los falsos negativos son 11 y los falsos positivos ascienden a 39. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 50.

El modelo generado en esta ejecución presenta el peor porcentaje de éxito de los presentados hasta el momento.

En la matriz de confusión se aprecia una cantidad de falsos negativos que no está entre las más altas. De hecho, si bien el modelo presentado anteriormente presentaba un menor porcentaje de error, la cantidad de falsos negativos entregados era mayor. Es decir, este modelo omitiría una menor cantidad de similitudes relevantes que el anterior.

El número de falsos positivos, en cambio, es el peor de todos los registrados hasta el momento. Ese número nos dice que, eventualmente, este modelo entregaría un mayor número de “falsas alarmas” que todos los modelos anteriores. Entonces, si bien el número de alarmas relevantes omitidas no es tanto en comparación con los otros modelos MLP generados, es el que peor cumpliría la labor de reducir las alarmas irrelevantes.

5.2.1.8 Octava ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	576	93.2039 %
Instancias mal clasificadas	42	6.7961 %

Tabla 8: Porcentajes de éxito y de error.

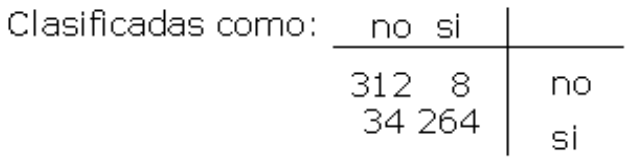


Figura 28: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 312 verdaderos negativos y 264 verdaderos positivos. Estas son las instancias bien clasificadas y suman 576.

Por otro lado, los falsos negativos son 34 y los falsos positivos ascienden a 8. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 42.

El modelo generado en esta ejecución presenta un porcentaje de éxito cercano a la media de casos registrados.

La cantidad de falsos positivos es la mejor registrada hasta el momento,

y con apreciable diferencia. Es decir, el modelo generado en esta ejecución sería el mejor en cuanto a eliminar falsas alarmas.

Sin embargo, así como la cantidad de falsos positivos es favorablemente baja, la cantidad de falsos negativos es, por amplia diferencia, la más alta registrada hasta el momento (más del doble de la peor registrada anteriormente), lo cual quiere decir que es, con amplio márgen, el modelo que más similitudes relevantes omite, lo cual es indeseable para la solución.

5.2.1.9 Novena ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	578	93.5275 %
Instancias mal clasificadas	40	6.4725 %

Tabla 9: Porcentajes de éxito y de error.

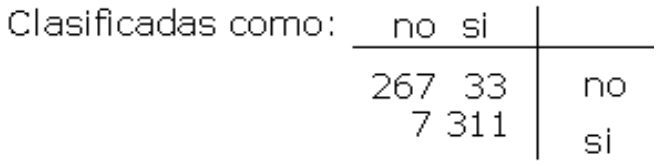


Figura 29: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 267 verdaderos negativos y 311 verdaderos positivos. Estas son las instancias bien clasificadas y suman 578.

Por otro lado, los falsos negativos son 7 y los falsos positivos ascienden a

33. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 40.

El modelo generado presenta un porcentaje de éxito aceptable en comparación con los anteriores.

La cantidad de falsos negativos es de las más bajas presentadas entre los modelos generados hasta el momento. Esto es, el modelo entregaría una cantidad comparativamente baja de similitudes irrelevantes.

La cantidad de falsos positivos es relativamente alta, de hecho es una de las más altas registradas. Con lo anterior, se puede decir que de este modelo se espera una cantidad relativamente baja de omisiones de similitudes relevantes, pero una cantidad comparativamente alta de similitudes irrelevantes. Este modelo podría ser considerado entre los más "permisivos": tiende a clasificar casos como similitudes (lo sean o no) con mayor frecuencia que otros de los modelos generados.

5.2.1.10 Décima ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	579	93.6893 %
Instancias mal clasificadas	39	6.3107 %

Tabla 10: Porcentajes de éxito y error.

```
Clasificadas como:  no  si  |
                    ------------
                    289  24 | no
                     15 290 | si
```

Figura 30: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a “no-similitud”.

Si corresponde a “similitud”.

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 289 verdaderos negativos y 290 verdaderos positivos. Estas son las instancias bien clasificadas y suman 579.

Por otro lado, los falsos negativos son 15 y los falsos positivos ascienden a 24. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 39.

El modelo generado presenta un porcentaje de éxito similar al anterior.

La cantidad de falsos negativos es relativamente alta dentro de los modelos MLP generados, lo cual implica una cantidad relativamente alta de similitudes relevantes omitidas (sin considerar el modelo generado en la octava ejecución).

La cantidad de falsos positivos no es de las más altas. Es decir, en comparación con los otros modelos generados, la cantidad de similitudes irrelevantes o falsas alarmas reportadas no es muy alta.

5.2.1.11 Resumen de modelos MLP.

Se presenta a continuación una tabla resumen de los resultados obtenidos en las pruebas con los modelos MLP generados.

Tamaño del set de entrenamiento	3858
Instancias de validación	618
Éxito:	
Promedio	93.61%
Desviación Estándar	0.95%
Mediana	93.45%
Error:	
Promedio	6.39%
Desviación Estándar	0.95%
Mediana	6.55%
Verdaderos Positivos:	
Promedio	283.30
Desviación Estándar	18.16
Mediana	289.50
Verdaderos Negativos:	
Promedio	295.20
Desviación Estándar	16.49
Mediana	291.00
Falsos Positivos:	
Promedio	26.10
Desviación Estándar	9.90
Mediana	26.00
Falsos Negativos:	
Promedio	13.40
Desviación Estándar	8.13
Mediana	11.00

Tabla 11: Resumen de resultados obtenidos en modelos MLP

Se debe notar que el porcentaje de éxito y el porcentaje de error varían poco, lo cual se puede apreciar por la desviación estándar de ambos valores, que es inferior al 1%.

Los errores de tipo I y II (falsos positivos y falsos negativos), en cambio, presentan variaciones mayores a lo largo de las diez ejecuciones, observándose desviaciones estándar más importantes para estos valores.

Junto al cuadro resumen, es interesante observar gráficamente cómo evolucionan estas cantidades a lo largo de las diez ejecuciones:

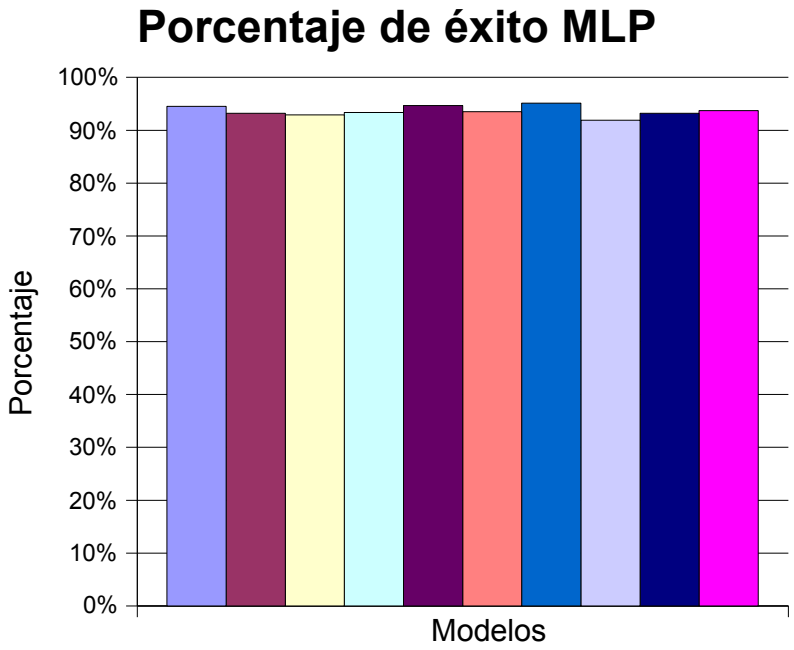


Figura 31: Histograma de evolución del porcentaje de éxito en los 10 modelos MLP generados.

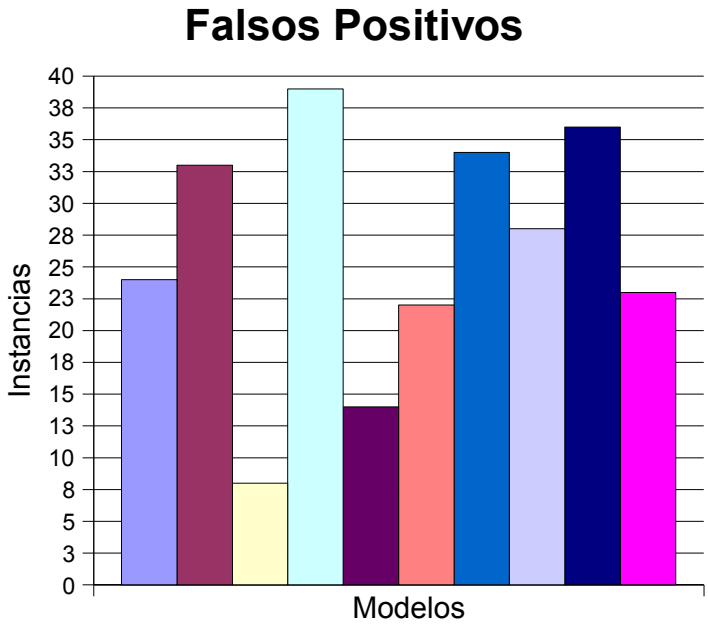


Figura 32: Cantidad de instancias correspondientes a error de tipo I en cada ejecución.

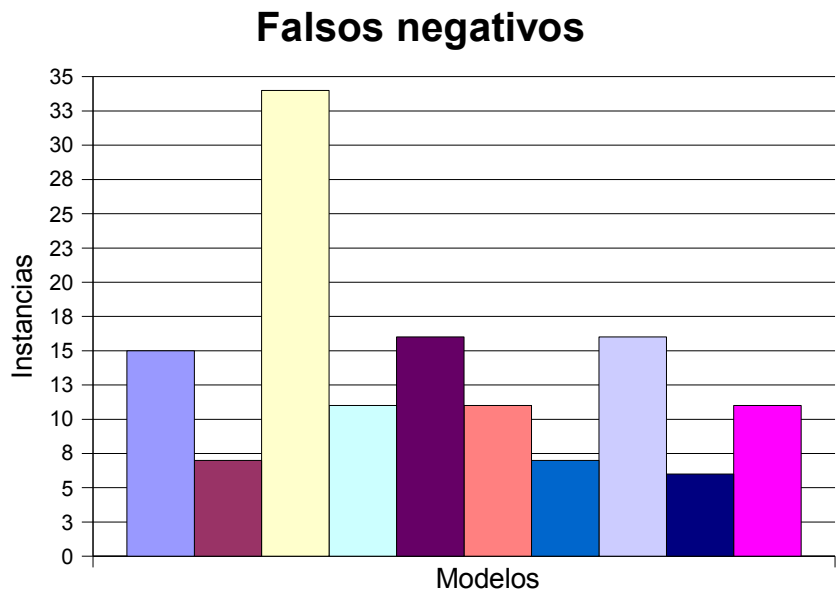


Figura 33: Instancias correspondientes a error de tipo II en cada ejecución.

Tal como las cantidades reportadas en el cuadro resumen (Tabla 11), los gráficos de las figuras 31,32 y 33 muestran de modo claro lo anteriormente planteado: el porcentaje de éxito presenta una variación leve en comparación con las instancias correspondientes a errores de tipo I y II. Esto es, los modelos generados, en general, cometen errores en números similares de instancias. Lo que varía de modo más considerable son los tipos de errores cometidos de uno a otro modelo.

5.2.2 SVM

Se entrenó diez veces el modelo para obtener un clasificador de tipo máquina de soporte vectorial (SVM), siempre utilizando el mismo conjunto de entrenamiento (que es, además, el mismo conjunto que se utilizó para entrenar el clasificador MLP). Los resultados del entrenamiento y validación fueron los siguientes:

5.2.2.1 Primera ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	593	95.9547 %
Instancias mal clasificadas	25	4.0453 %

Tabla 12: Porcentajes de éxito y de error.

Clasificadas como:	no	si	
	297	19	no
	6	296	si

Figura 34: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 297 verdaderos negativos y 296 verdaderos positivos. Estas son las instancias bien clasificadas y suman 593.

Por otro lado, los falsos negativos son 6 y los falsos positivos ascienden a 19. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 25.

De este modo, se aprecia que, en esta ejecución, el porcentaje de instancias bien clasificadas es alto, cercano al 96% (superior a cualquiera de los modelos MLP generados).

La tabla de decisión muestra seis falsos negativos, cantidad que compara, también, muy favorablemente con los modelos MLP generados. Es decir, el modelo SVM generado omitiría menos similitudes relevantes que el promedio de los modelos MLP y, de hecho, menos que casi todos ellos.

Por otra parte, como en la mayoría de los casos de modelos MLP vistos, hay más falsos positivos que falsos negativos. Eso es lo esperable, porque la omisión de casos relevantes es un error más crítico que que el reporte de casos irrelevantes. Además de esto, el número de falsos positivos obtenidos compara, también, favorablemente con la mayoría de los modelos MLP obtenidos.

5.2.2.2 Segunda ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	579	93.6893 %
Instancias mal clasificadas	39	6.3107 %

Tabla 13: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                    ------------
                    321  23 | no
                     16 258 | si
```

Figura 35: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 321 verdaderos negativos y 258 verdaderos positivos. Estas son las instancias bien clasificadas y suman 579.

Por otro lado, los falsos negativos son 16 y los falsos positivos ascienden a 23. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 39.

Se puede apreciar que el modelo generado en esta segunda ejecución presenta un mayor porcentaje de error que el modelo generado en la primera.

La tabla de decisión muestra que también aumenta el número de falsos negativos. Es decir, el modelo eventualmente omitiría más similitudes relevantes que el anterior.

Se aprecia, por otro lado, que los falsos positivos aumentaron también, pero a una tasa menor que los falsos negativos. Es decir, la proporción falsos positivos/falsos negativos disminuyó. Lo que esto quiere decir, es que la diferencia entre las falsas alarmas y las similitudes omitidas es menor que en el caso anterior.

Los resultados obtenidos con este modelo son muy comparables a los obtenidos, en promedio, con los modelos MLP.

5.2.2.3 Tercera ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	585	94.6602 %
Instancias mal clasificadas	33	5.3398 %

Tabla 14: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                    ------------
                   299  23  |  no
                    10 286  |  si
```

Figura 36: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 299 verdaderos negativos y 286 verdaderos positivos. Estas son las instancias bien clasificadas y suman 585.

Por otro lado, los falsos negativos son 10 y los falsos positivos ascienden a 23. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 33.

Este modelo presenta un mejor porcentaje de éxito que el caso anterior y que la media de los modelos MLP.

La tabla de decisión muestra una cantidad de falsos negativos que también compara positivamente con el caso anterior y con los modelos MLP. No tan favorable, sin embargo, comparado con el primer modelo SVM.

En cuanto a los falsos positivos, es el mismo número que en el caso anterior. Comparando, es posible decir que este modelo SVM se asemeja a algunos de los mejores modelos obtenidos con MLP, en cuanto a porcentaje de error y a cantidad de errores de cada tipo. De todos modos, este modelo omite más similitudes relevantes y entrega más falsas alarmas que el primer SVM generado, si bien compara favorablemente con el segundo y con los modelos MLP.

5.2.2.4 Cuarta ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	587	94.9838 %
Instancias mal clasificadas	31	5.0162 %

Tabla 15: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                   ---------+
                   288  23  | no
                     8 299  | si
```

Figura 37: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 288 verdaderos negativos y 299 verdaderos positivos. Estas son las instancias bien clasificadas y suman 587.

Por otro lado, los falsos negativos son 8 y los falsos positivos ascienden a 23. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman

31.

Este modelo presenta resultados bastante similares al modelo inmediatamente anterior. En efecto, el porcentaje de error es muy parecido.

La tabla de decisión muestra que tanto los errores de tipo I como de tipo II son similares al caso anterior.

Lo anterior significa que este modelo entrega un número muy similar de instancias erróneamente clasificadas, pero además, que reporta un número similar de casos: omite una cantidad parecida de similitudes relevantes y reporta una cantidad parecido de similitudes irrelevantes. Los números son muy levemente mejores.

De este modo, este modelo compara favorablemente con los otros modelos SVM generados (sin alcanzar el nivel de éxito del primer modelo generado), y compara favorablemente con los modelos MLP generados.

5.2.2.5 Quinta ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	587	94.9838 %
Instancias mal clasificadas	31	5.0162 %

Tabla 16: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                   ---------+
                    280  20 | no
                     11 307 | si
```

Figura 38: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a “similitud”.

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 280 verdaderos negativos y 307 verdaderos positivos. Estas son las instancias bien clasificadas y suman 587.

Por otro lado, los falsos negativos son 11 y los falsos positivos ascienden a 20. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 31.

La validación de este modelo presenta exactamente el mismo porcentaje de error que el modelo anterior.

Sin embargo, la matriz de confusión es distinta. Este modelo presenta una cantidad algo mayor de falsos negativos, es decir, omite más similitudes relevantes que el caso anterior.

En cambio, los falsos positivos aumentan en el mismo número. Es decir, hay un aumento en las similitudes irrelevantes, pero de la misma forma disminuyen las falsas alarmas. Los números son similares en ambos casos, la diferencia es que el cuarto modelo es “menos estricto” que el quinto, es decir, tiende a descartar menos similitudes. El modelo quinto, en cambio, es levemente más “permisivo”, en el sentido de que deja pasar algunos casos más como similitudes.

Este modelo compara favorablemente con los modelos MLP generados, tanto en porcentaje de éxito como en cuanto a falsos negativos, pero especialmente en cuanto a falsos positivos (este modelo arrojaría menos falsas alarmas).

5.2.2.6 Sexta ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	582	94.1748 %
Instancias mal clasificadas	36	5.8252 %

Tabla 17: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                   ---------+
                    322  26 | no
                     10 260 | si
```

Figura 39: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 322 verdaderos negativos y 260 verdaderos positivos. Estas son las instancias bien clasificadas y suman 582.

Por otro lado, los falsos negativos son 10 y los falsos positivos ascienden a 26. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 36.

La validación de este modelo presenta resultados cercanos a los anteriores. Hasta el momento se ha visto poca variación en el porcentaje de error de los modelos SVM.

La matriz de confusión revela un número muy similar de falsos negativos a los presentados por el modelo anterior. Es decir, un grado de omisión muy

similar de similitudes relevantes.

La diferencia principal está en la cantidad de falsos positivos, que en este caso es mayor. Es decir, este caso omite la misma cantidad de similitudes irrelevantes que el caso anterior, pero arroja más falsas alarmas, lo cual es evidentemente un resultado menos favorable.

El caso compara, de todos modos, de modo relativamente favorable con los modelos MLP generados, siendo un poco mejor que la media de aquellos.

5.2.2.7 Séptima ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	588	95.1456 %
Instancias mal clasificadas	30	4.8544 %

Tabla 18: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                   ---------+
                   303  17  | no
                    13 285  | si
```

Figura 40: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 303 verdaderos negativos y 285 verdaderos positivos. Estas son las instancias bien clasificadas y suman 588.

Por otro lado, los falsos negativos son 13 y los falsos positivos ascienden a 13. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman

30.

La validación de este modelo presenta resultados que se acercan a los anteriores, pero resultan levemente superiores (excepto al primero).

La matriz de confusión revela un número un poco mayot de falsos negativos a los presentados por el modelo anterior. Es decir, un grado de omisión un poco superior de similitudes relevantes.

Sin embargo, la principal diferencia se marca en cuanto a la cantidad de falsos positivos, en que este modelo entrega un número más favorable.

Este modelo, de hecho, en la validación reporta un número menor de falsas alarmas que todos los otros modelos SVM generados hasta el momento.

El caso compara de modo muy favorable contra los modelos MLP generados, estando por sobre la media y entregando mejores números en los errores de tipo I y II.

5.2.2.8 Octava ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	588	95.1456 %
Instancias mal clasificadas	30	4.8544 %

Tabla 19: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                   ---------+
                   289  24  | no
                     6 299  | si
```

Figura 41: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a “no-similitud”.

Si corresponde a “similitud”.

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 289 verdaderos negativos y 299 verdaderos positivos. Estas son las instancias bien clasificadas y suman 588.

Por otro lado, los falsos negativos son 6 y los falsos positivos ascienden a 24. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 30.

La validación de este modelo presenta exactamente el mismo porcentaje de error que el caso anterior.

La matriz de confusión es lo que varía. Es decir, el error en la clasificación es lo mismo, la diferencia está en cuánto de ese error corresponde a error de tipo I o tipo II.

La cantidad de falsos negativos disminuye. Es decir, el modelo omite menos similitudes relevantes que el caso anterior.

Los falsos positivos, en cambio, aumentan, lo que significa que este modelo, si bien omite menos alertas relevantes, es menos efectivo en controlar el reporte de similitudes irrelevantes.

El caso compara de modo favorable contra los modelos MLP generados, estando por sobre la media y entregando mejores números en cuanto a porcentaje de error, notablemente mejor resultando en cuanto a omisión de similitudes relevantes, y un número levemente inferior de falsas alarmas.

5.2.2.9 Novena ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	585	94.6602 %
Instancias mal clasificadas	33	5.3398 %

Tabla 20: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                   ---------+
                    280  20 | no
                     13 305 | si
```

Figura 42: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 280 verdaderos negativos y 305 verdaderos positivos. Estas son las instancias bien clasificadas y suman 585.

Por otro lado, los falsos negativos son 13 y los falsos positivos ascienden a 20. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 33. La validación de este modelo presenta exactamente el mismo porcentaje de error que el tercer modelo SVM generado.

La matriz de confusión es levemente distinta. Es decir, tal como se explicaba para el caso anterior, el error en la clasificación es lo mismo, la diferencia está en cuánto de ese error corresponde a error de tipo I o tipo II.

La cantidad de falsos negativos aumenta. El modelo omite un número

superior de similitudes relevantes que el tercer modelo.

Los falsos positivos, por el contrario, disminuyen (en el mismo número que los falsos negativos aumentan), lo que significa que este modelo omite algunas similitudes relevantes más que el otro modelo, pero por otra parte entrega algunas falsas alarmas menos.

El caso compara favorablemente con los modelos MLP. Presenta porcentajes de errores de ambos tipos más favorables que varios de ellos, y los números son algo superiores a la media de los mismos.

5.2.2.10 Décima ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	584	94.4984 %
Instancias mal clasificadas	34	5.5016 %

Tabla 21: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                   ---------+
                   287  26  | no
                     8 297  | si
```

Figura 43: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 287 verdaderos negativos y 297 verdaderos positivos. Estas son las instancias bien clasificadas y suman 584.

Por otro lado, los falsos negativos son 8 y los falsos positivos ascienden a 26. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 34.

La validación de este modelo presenta un porcentaje de error bastante cercano al caso presentado anteriormente.

La matriz de confusión muestra, por otra parte, un número inferior de falsos negativos, lo cual quiere decir que durante el período de validación, el modelo cometió un número menor de errores de tipo II: omitió menos similitudes relevantes.

La cantidad de falsos positivos, en cambio, es alta. En efecto, entre los modelos SVM generados es uno de los peores en ese sentido.

El caso presenta ciertas similitudes con los modelos MLP. El porcentaje de error supera a la media de aquellos por un valor cercano al 0,9%, mientras la cantidad de errores de tipo I es la misma.

En cuanto a error de tipo II, en cambio, el modelo presenta un mejor resultado que la media de los modelos MLP, y mejores resultados que la mayoría de aquellos.

5.2.2.11 Resumen de modelos SVM.

Se presenta a continuación una tabla resumen de los resultados obtenidos en las pruebas con los modelos SVM generados.

Tamaño del set de entrenamiento	3858
Instancias de validación	618
Éxito:	
Promedio	94.79%
Desviación Estándar	0.61%
Mediana	94.82%
Error:	
Promedio	5.21%
Desviación Estándar	0.61%
Mediana	5.18%
Verdaderos Positivos:	
Promedio	289.20
Desviación Estándar	17.38
Mediana	296.50
Verdaderos Negativos:	
Promedio	296.60
Desviación Estándar	15.14
Mediana	293.00
Falsos Positivos:	
Promedio	22.10
Desviación Estándar	3.00
Mediana	23.00
Falsos Negativos:	
Promedio	10.10
Desviación Estándar	3.25
Mediana	10.00

Tabla 22: Resumen de resultados obtenidos en modelos SVM

Se debe notar que el porcentaje de éxito y el porcentaje de error varían poco, lo cual se puede apreciar por la desviación estándar de ambos valores, iguales al 0,61%. Así, también, se aprecia que estos valores varían menos que en el caso de los modelos MLP, en que se observaba una desviación estándar

igual al 0,95%.

Los errores de tipo I y II (falsos positivos y falsos negativos) también varían menos que en el caso de los modelos MLP: en el caso SVM ambas desviaciones estándar están alrededor de 3, mientras para MLP eran totales superiores a las 8 instancias. Esto da cuenta de que los diez modelos SVM son mucho más similares entre sí que los modelos MLP generados.

Junto al cuadro resumen, es interesante observar gráficamente cómo evolucionan estas cantidades a lo largo de las diez ejecuciones:

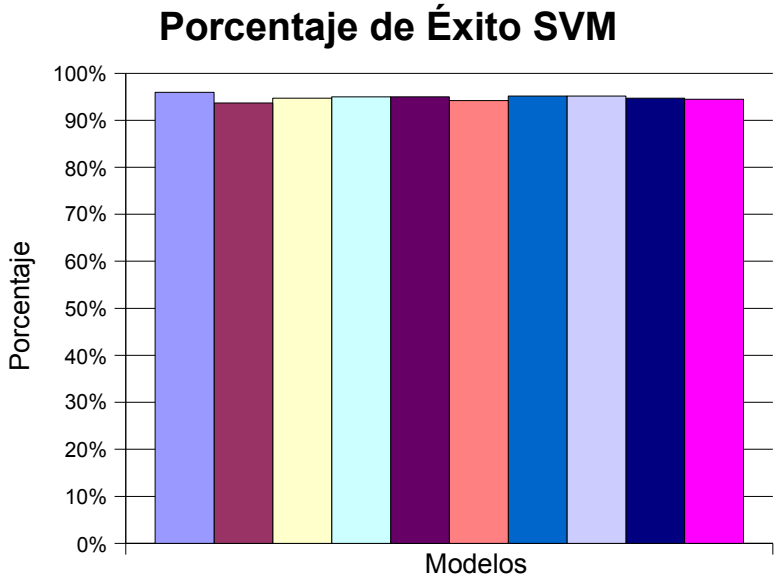


Figura 44: Histograma de evolución del porcentaje de éxito en los 10 modelos SVMgenerados.

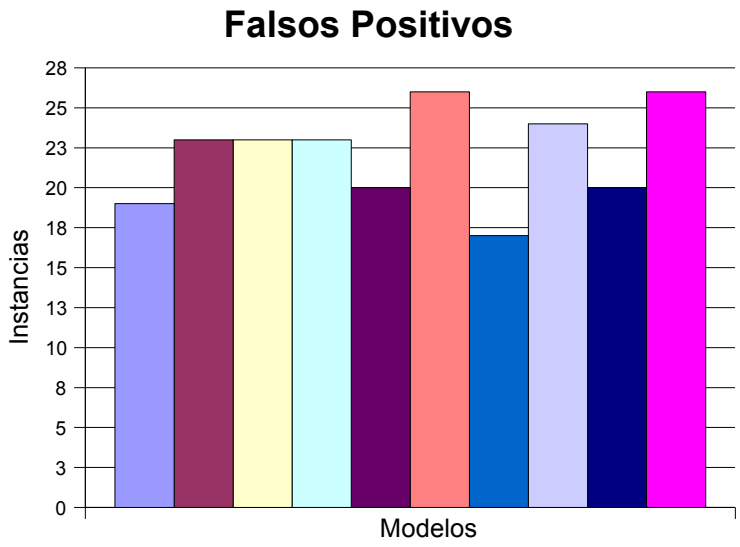


Figura 45: Cantidad de instancias correspondientes a error de tipo I en cada ejecución.

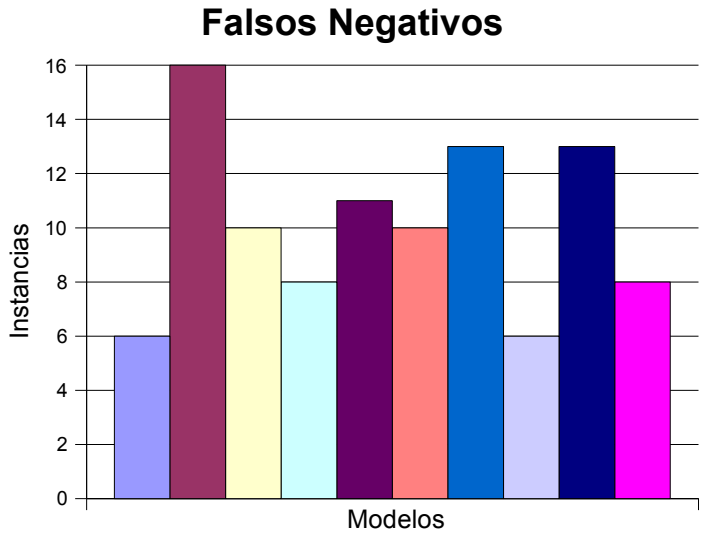


Figura 46: Instancias correspondientes a error de tipo II en cada ejecución.

Tal como las cantidades reportadas en el cuadro resumen (Tabla 22), los gráficos de las figuras 44,45 y 46 permiten apreciar de modo claro lo que se planteaba con respecto a las similitudes de todos estos modelos, además de las diferencias con el caso MLP.

Se debe mencionar, además, que el porcentaje de éxito promedio de los modelos SVM es superior en 1.18% al promedio de los modelos MLP.

Además, se puede apreciar que los valores correspondientes a errores de tipo I y II son, por separado, mejores que los valores obtenidos en modelos MLP.

5.2.3 PART

Se entrenó diez veces el modelo para obtener un clasificador de tipo tabla de decisión PART, siempre utilizando el mismo conjunto de entrenamiento (que es, además, el mismo conjunto que se utilizó para entrenar los clasificadores anteriores). Los resultados del entrenamiento y validación fueron los siguientes:

5.2.3.1 Primera ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	596	96.4401 %
Instancias mal clasificadas	36	5.8252 %

Tabla 23: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                   ---------+
                   301  15  | no
                     7 295  | si
```

Figura 47: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 301 verdaderos negativos y 295 verdaderos positivos. Estas

son las instancias bien clasificadas y suman 596.

Por otro lado, los falsos negativos son 7 y los falsos positivos ascienden a 15. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 22.

De este modo, se aprecia que, en esta ejecución, el porcentaje de instancias bien clasificadas es alto, superior al 96%, lo cual es un mejor porcentaje que el obtenido con todos los modelos generados anteriormente, ya sea MLP o SVM.

La tabla de decisión muestra una mayor cantidad de falsos positivos que de falsos negativos. Eso es normal, considerando que omitir similitudes relevantes es un error más crítico que reportar similitudes irrelevantes. Comparativamente con los modelos MLP y SVM generados, la cantidad de falsos negativos que se obtiene con este modelo resulta bastante conveniente, por ser un número bajo de errores de este tipo.

También es relativamente bajo el número de falsos positivos. Por lo tanto, además de omitir pocas similitudes relevantes, este modelo entregaría un número bajo de falsas alarmas.

5.2.3.2 Segunda ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	582	94.1748 %
Instancias mal clasificadas	36	5.8252 %

Tabla 24: Porcentajes de éxito y de error.

Clasificadas como:	no	si	
	322	22	no
	14	260	si

Figura 48: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 322 verdaderos negativos y 260 verdaderos positivos. Estas son las instancias bien clasificadas y suman 582.

Por otro lado, los falsos negativos son 14 y los falsos positivos ascienden a 22. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 36.

De este modo, se aprecia que, en esta ejecución, el porcentaje de instancias clasificadas correctamente es poco superior al 94%, lo cual es un resultado aceptable comparado con modelos anteriores. En general, los modelos SVM entregan mejores resultados (el promedio es más alto), pero es superior a los modelos MLP. El modelo PART anterior, de todos modos, entrega un mejor porcentaje.

La tabla de decisión muestra una mayor cantidad de falsos positivos que de falsos negativos, pero en una diferencia más moderada de la que se apreciaba en el modelo anterior. Los falsos positivos aumentan, y se observa un resultado menos conveniente, incluso, que el promedio de los modelos MLP.

En cuanto a los falsos positivos, estos también empeoran en relación al caso anterior. La cantidad de falsos positivos para este modelo resulta, de todos modos, similar al promedio de los falsos positivos de modelos SVM y menor que el promedio de falsos positivos de modelos MLP. Es decir, el modelo

entregaría una cantidad aceptable de falsas alarmas en relación a los modelos previamente mostrados.

5.2.3.3 Tercera ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	578	93.5275 %
Instancias mal clasificadas	40	6.4725 %

Tabla 25: Porcentajes de éxito y de error.

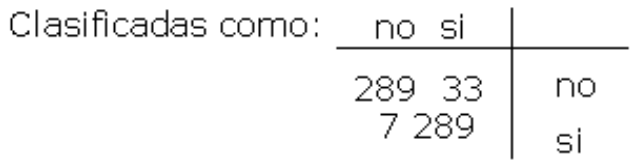


Figura 49: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 289 verdaderos negativos y 289 verdaderos positivos. Estas son las instancias bien clasificadas y suman 578.

Por otro lado, los falsos negativos son 7 y los falsos positivos ascienden a 33. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 40.

En esta ejecución, puede observarse, el porcentaje de instancias bien

clasificadas es peor que el promedio de los modelos SVM e incluso inferior al porcentaje de éxito promedio de los modelos MLP (aunque muy similar).

El detalle de la tabla de confusión permite apreciar que la cantidad de falsos negativos es baja en comparación con los modelos anteriores (de los tres tipos). Este aspecto es positivo, porque este modelo omitiría una cantidad inferior de similitudes relevantes baja, menor que la mayoría de los modelos anteriores.

En cuanto a los falsos positivos, estos presentan una cantidad muy elevada con respecto a los modelos de los tres tipos presentados anteriormente. Por lo tanto, este modelo sería comparativamente malo para el propósito de evitar las falsas alarmas.

5.2.3.4 Cuarta ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	585	94.6602 %
Instancias mal clasificadas	33	5.3398 %

Tabla 26: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                    ------------
                   298  13  | no
                    20 287  | si
```

Figura 50: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 298 verdaderos negativos y 287 verdaderos positivos. Estas son las instancias bien clasificadas y suman 585.

Por otro lado, los falsos negativos son 20 y los falsos positivos ascienden a 13. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 33.

El porcentaje de instancias bien clasificadas en este caso es muy similar a la media de los casos SVM. Es, por tanto, mejor que los modelos MLP y, aunque mejor que los dos modelos PART presentados anteriormente, es peor que el primero.

La matriz de confusión permite observar una inversión con respecto a la mayoría de los casos anteriores, en el sentido de que los falsos negativos son más que los falsos positivos. Es decir, el modelo presentaría un número mayor de omisiones que de falsas alarmas, lo cual no es deseable, especialmente considerando que este número de omisiones es alto en comparación al promedio de casos presentados anteriormente.

Con respecto a las falsas alarmas (falsos positivos), este es uno de los números más favorables apreciados hasta el momento.

5.2.3.5 Quinta ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	579	93.6893 %
Instancias mal clasificadas	39	6.3107 %

Tabla 27: Porcentajes de éxito y de error.

Clasificadas como:	no	si	
	282	18	no
	21	297	si

Figura 51: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 282 verdaderos negativos y 297 verdaderos positivos. Estas son las instancias bien clasificadas y suman 579.

Por otro lado, los falsos negativos son 21 y los falsos positivos ascienden a 18. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 39.

El porcentaje de instancias bien clasificadas en este caso es similar a la media de los casos MLP. Es, por tanto, inferior al resultado obtenido para los modelos SVM.

La matriz de confusión revela que en este cas nuevamente los falsos positivos son menos que los falsos negativos. Es decir, el modelo presentaría un número alto de omisiones (en efecto, aún mayor que en el caso anterior). Este es uno de los peores resultados obtenidos en este sentido.

Los falsos positivos obtenidos son más que en el caso anterior, pero aún inferior al promedio de falsos positivos para la media de los modelos SVM obtenidos.

5.2.3.6 Sexta ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	596	96.4401 %
Instancias mal clasificadas	22	3.5599 %

Tabla 28: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                    ------------
                    316  10 | no
                     12 280 | si
```

Figura 52: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 316 verdaderos negativos y 280 verdaderos positivos. Estas son las instancias bien clasificadas y suman 596.

Por otro lado, los falsos negativos son 12 y los falsos positivos ascienden a 10. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 22.

El porcentaje de instancias bien clasificadas en este caso es , junto con el primer modelo PART presentado, el mejor de todos los modelos generados hasta el momento. Un porcentaje de éxito superior al 96%.

A diferencia del primer modelo PART generado, en este caso los falsos negativos son más que los falsos positivos, aunque las cantidades de uno y otro tipo de error son similares: 12 contra 10. El número de falsos negativos, de todos modos, si bien es mayor que el número de falsos positivos, aún compara favorablemente con la media de los casos presentados previamente.

Los falsos positivos presentados en este caso son, como se dijo, menos que los falsos negativos, y menos que en la mayoría de los casos presentados anteriormente. Este modelo es, dentro de todos los presentados hasta el momento, uno de los mejores en la búsqueda del control de falsas alarmas.

5.2.3.7 Séptima ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	578	93.5275 %
Instancias mal clasificadas	40	6.4725 %

Tabla 29: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                   ---------+
                   320  28  | no
                    12 258  | si
```

Figura 53: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se

aprecia un total de 320 verdaderos negativos y 258 verdaderos positivos. Estas son las instancias bien clasificadas y suman 578.

Por otro lado, los falsos negativos son 12 y los falsos positivos ascienden a 28. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 40.

El porcentaje de instancias bien clasificadas en este caso es comparativamente bajo, si se mira en relación a los modelos SVM y a los PART más exitosos. Es muy similar a la media de los modelos MLP.

Además, el detalle de la matriz de confusión corrobora el parecido con la media de los modelos MLP. La cantidad de falsos negativos es cercana al promedio de modelos MLP, es decir, omite una cantidad similar de alarmas relevantes.

Así también los falsos positivos son comparables a la media de modelos MLP, por lo cual este modelo presentaría una cantidad similar de falsas alarmas.

5.2.3.8 Octava ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	588	95.1456 %
Instancias mal clasificadas	30	4.8544 %

Tabla 30: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                    ------------
                    304  16 | no
                     14 284 | si
```

Figura 54: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a “no-similitud”.

Si corresponde a “similitud”.

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 304 verdaderos negativos y 284 verdaderos positivos. Estas son las instancias bien clasificadas y suman 588.

Por otro lado, los falsos negativos son 14 y los falsos positivos ascienden a 16. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 30.

El porcentaje de instancias bien clasificadas, si bien es algo inferior a los mejores modelos PART presentados, es alto en comparación con otros modelos PART, además de los modelos MLP y SVM.

En la matriz de confusión se aprecia un grado de equilibrio en la cantidad de errores de cada tipo. La cantidad de falsos negativos es cercana a la media de los casos presentados anteriormente (en modelos de distintos tipos).

Por otra parte, la cantidad de falsos positivos, si bien no es tan favorable como en otros modelos PART presentados, compara de modo muy adecuado con la media de casos. En efecto, es inferior tanto a la media de modelos MLP como a la media de modelos SVM.

Por lo tanto, este modelo presentaría una cantidad cercana a la media de omisiones, pero por otro lado, sería más efectivo que la mayoría de los modelos en el control o eliminación de falsas alarmas.

5.2.3.9 Novena ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	586	94822,00%
Instancias mal clasificadas	32	5178,00%

Tabla 31: Porcentajes de éxito y de error.

```
Clasificadas como:  no  si  |
                    ------------
                    297  16 | no
                     16 289 | si
```

Figura 55: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 297 verdaderos negativos y 286 verdaderos positivos. Estas son las instancias bien clasificadas y suman 586.

Por otro lado, los falsos negativos son 16 y los falsos positivos ascienden a 16. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 32.

Este modelo presenta resultados muy similares al anterior. El porcentaje de instancias bien clasificadas es levemente inferior. En el universo en que el modelo se validó, el modelo cometió dos errores de clasificación más que el anterior.

En la matriz de confusión se aprecia un exacto equilibrio en la cantidad de errores de cada tipo. El modelo, para el conjunto de validación, presentó exactamente el mismo número de omisiones como de falsas alarmas.

La cantidad de falsos negativos es superior a la media de los casos anteriormente reportados. Es decir, este modelo tendería a omitir algunos casos más que el promedio.

Con respecto a los falsos positivos, en cambio, el número es inferior a la media de los casos presentados anteriormente. Es decir, este modelo resultaría comparativamente efectivo para evitar las falsas alarmas.

5.2.3.10 Décima ejecución.

Sobre un conjunto total de 3858 instancias, se entrenó el modelo en 3240 y se validó en las otras 618, obteniéndose:

Instancias bien clasificadas	583	94.3366 %
Instancias mal clasificadas	35	5.6634 %

Tabla 32: Porcentajes de éxito y de error.

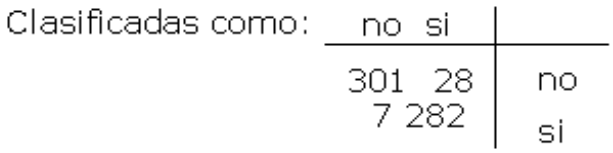


Figura 56: Matriz de confusión.

En la matriz de confusión presentada:

No corresponde a "no-similitud".

Si corresponde a "similitud".

De esta matriz, y definiendo "si" como positivo y "no" como negativo, se aprecia un total de 301 verdaderos negativos y 282 verdaderos positivos. Estas son las instancias bien clasificadas y suman 583.

Por otro lado, los falsos negativos son 7 y los falsos positivos ascienden a 28. Las instancias mal clasificadas (falsos positivos y falsos negativos) suman 35.

Este modelo presenta un porcentaje de instancias bien clasificadas que se asemeja a los modelos SVM. Es superior al promedio de los modelos MLP.

La matriz de confusión revela que el mejor aspecto de este modelo, es que presentaría pocas omisiones de similitudes relevantes. Los falsos negativos, como se puede apreciar, representan un número bajo en comparación con los modelos presentados anteriormente.

Los falsos positivos, en cambio, constituyen un número comparativamente alto, por lo cual este modelo no resultaría de los mejores en el propósito de evitar falsas alarmas.

5.2.3.11 Resumen de modelos PART

Tamaño del set de entrenamiento	3858
Instancias de validación	618
Éxito:	
Promedio	94.68%
Desviación Estándar	1.08%
Mediana	94.50%
Error:	
Promedio	5.32%
Desviación Estándar	1.08%
Mediana	5.50%
Verdaderos Positivos:	
Promedio	282.10
Desviación Estándar	13.27
Mediana	285.50
Verdaderos Negativos:	
Promedio	303.00
Desviación Estándar	13.02
Mediana	301.00
Falsos Positivos:	
Promedio	19.90
Desviación Estándar	7.53
Mediana	17.00
Falsos Negativos:	
Promedio	13.00
Desviación Estándar	5.10
Mediana	13.00

Tabla 33: Resumen de resultados obtenidos en modelos SVM

Se presenta a continuación una tabla resumen de los resultados obtenidos en las pruebas con los modelos PART generados. De los tres tipos de modelos presentados, los PART presentan los resultados más variables de un modelo a otro, en lo que a porcentaje de éxito y de error se refiere, y así lo muestra la desviación estándar, superior al 1%. Además, se aprecia que en promedio, estos modelos tienen mayor porcentaje de error que los modelos SVM, peor mayor que los MLP.

Con respecto a los errores de tipo I y II, estos (a diferencia de lo que ocurre con el porcentaje de error) varían poco en comparación con los otros tipos de modelos. En efecto, presentan menor desviación estándar.

El promedio de errores de tipo I es bueno comparado con los otros tipos de modelos. Es, efectivamente, el menor de los obtenidos. El promedio de errores de tipo II, en cambio, es mayor que para los modelos SVM y menor (levemente) que para los modelos MLP.

Junto al cuadro resumen, es interesante observar gráficamente cómo evolucionan estas cantidades a lo largo de las diez ejecuciones:

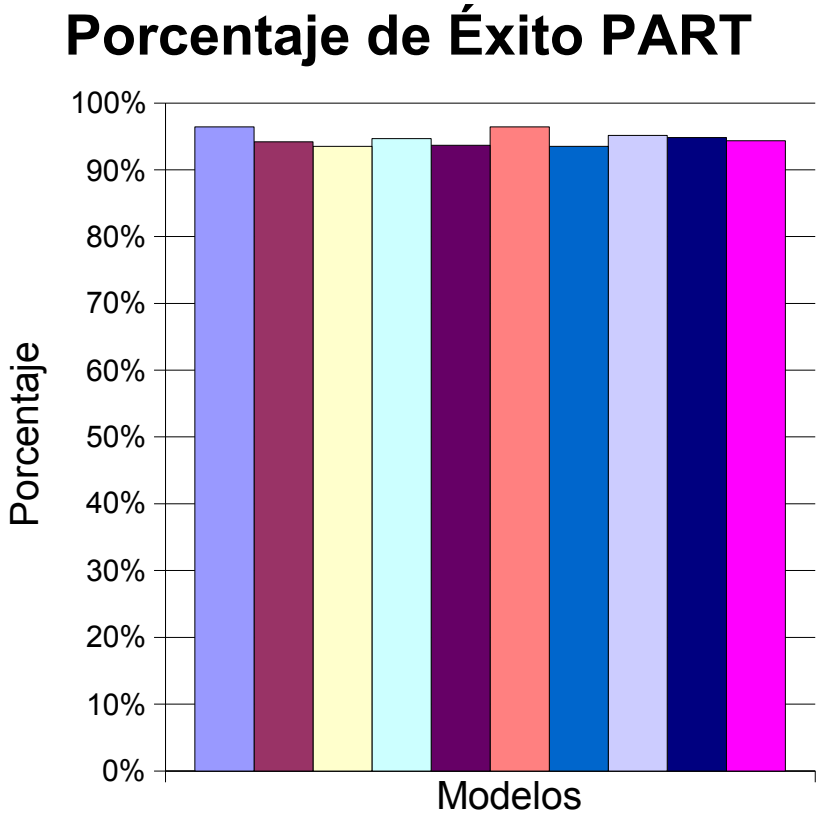


Figura 57: Histograma de evolución del porcentaje de éxito en los 10 modelos PART generados.

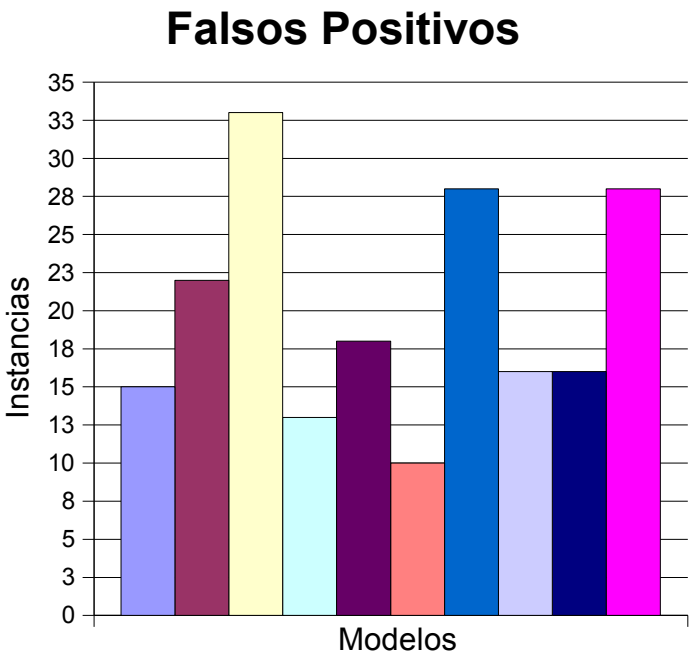


Figura 58: Cantidad de instancias correspondientes a error de tipo I en cada ejecución.

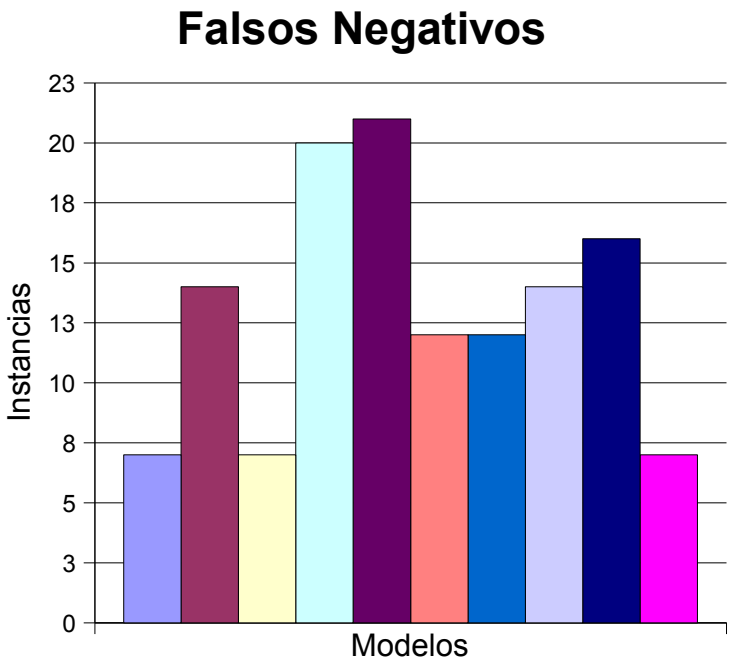


Figura 59: Instancias correspondientes a error de tipo II en cada ejecución.

Tal como las cantidades reportadas en el cuadro resumen (Tabla 33), los

gráficos de las figuras 57,58 y 59 permiten observar el comportamiento de estos modelos.

Es necesario apuntar que, si bien los resultados de los modelos PART son variables y tienen peor promedio que los modelos SVM, los mejores modelos PART obtenidos son mejores que todos los modelos MLP y SVM.

5.3 Comparación de modelos.

Como se puede ver en los cuadros presentados anteriormente, el tipo de modelo con el cual se obtiene, en promedio, el mejor resultado aparente es SVM, entendiendo como "mejor" el modelo que presenta menores porcentajes de error de tipo I y tipo II. Los resultados de estos modelos son parecidos entre sí, lo que se refleja en su desviación estándar menor que los otros casos.

Los modelos PART presentan, en promedio, porcentajes de error algo superiores y con mayor desviación estándar: los resultados son muy dispares de uno a otro modelo PART generado.

Por esto mismo, si bien algunos de los modelos PART generados presentan mayor cantidad de errores que la generalidad de los casos de modelos SVM (y similares a los modelos MLP), también existen casos, debido a los valores de desviación estándar mostrados anteriormente, de modelos PART que presentan resultados más favorables en cuanto a cantidad de errores que todos los modelos SVM logrados a lo largo de los diez procesos de entrenamiento.

Sin embargo, las diferencias porcentuales del error predicho rondan el 2% solamente, por lo cual se procederá a realizar el siguiente análisis:

- Se escoge el mejor modelo PART obtenido, es decir, el que presenta un menor porcentaje de error global, número inferior de errores de tipo I y un número comparativamente favorable de errores de tipo II. De este

modo, se escoge el primer modelo PART obtenido, con un error igual al 3,5599%, y que además presenta cantidades razonables de errores de tipo I y II (la cantidad de falsos negativos es comparativamente baja, y está en relación 1:2,14 con los falsos positivos). Considerando que es más urgente limitar los falsos negativos, para no omitir similitudes relevantes (y, por lo tanto, prestar un mejor servicio a los clientes), pero a la vez se debe mantener una cantidad razonable de falsos negativos, de modo que el costo de solución tenga sentido, se acepta este resultado.

- Se escoge el primer modelo SVM, que presenta un porcentaje de error comparativamente favorable (4,0453%).Además, los falsos negativos son relativamente bajos, y están en relación de 1:3,17 con los falsos positivos. Se tiene un menor número de falsos negativos que en el modelo PART (el modelo acusaría un mayor número de casos), pero a la vez el número de falsos positivos es mayor, por lo que se espera que el modelo reporte un mayor número de similitudes irrelevantes. En comparación con los otros modelos obtenidos, la relación falsos negativos a falsos positivos también es adecuada en este caso, por lo cual se acepta este modelo.

En realidad, a priori los modelos de los tres tipos evaluados presentan resultados muy similares en porcentaje. No es posible afirmar que alguno de ellos sea, definitivamente, mejor que los otros. En efecto, las diferencias de porcentaje de error son comparables con las desviaciones estándar.

En este contexto, el motivo por el cual no se escogió un modelo MLP para las pruebas siguientes, es que se optó por dos modelos puntuales entre aquellos generados de estos tres tipos que prometieran resultados favorables.

Una vez seleccionados ambios modelos, se deberá realizar pruebas para verificar cuál tiene una mejor performance en la práctica.

Se procederá a mostrar los resultados de la prueba de ambos modelos utilizando datos reales.

Comparando las curvas ROC (Receiver Operating Characteristic. En [4] se puede encontrar una amplia descripción de esta técnica y su interés en aplicaciones de minería de datos) de ambos modelos, se aprecian los resultados mostrados en las figuras 60 y 61, y que se comentan a continuación.

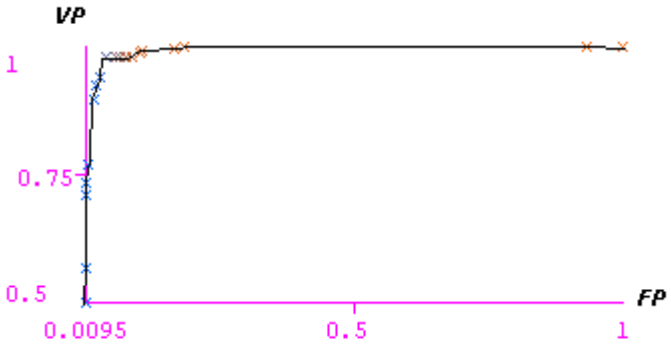


Figura 60: Curva ROC modelo PART

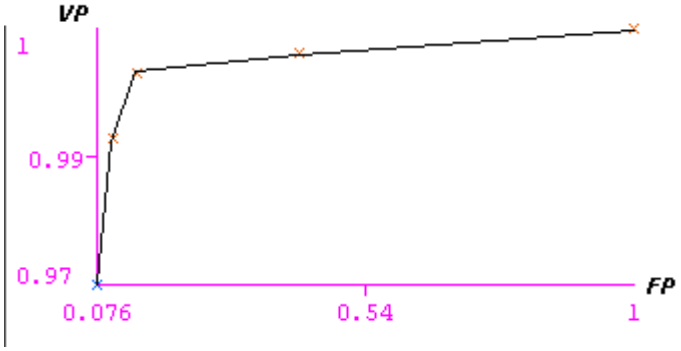


Figura 61: Curva ROC modelo SVM

Para el modelo PART, el área bajo la curva es 0.9845. Para el modelo SVM el valor es 0.9565.

Una forma de comparar dos clasificadores es mediante la comparación de sus curvas ROC. En efecto, el área bajo la curva es un parámetro importante de comparación.

En este caso, se aprecia que el área bajo la curva del clasificador PART es más cercano al clasificador ideal (área bajo la curva igual 1) que el del clasificador SVM.

Al margen de la comparación, en la curva ROC es interesante observar que la curva presenta verdaderos contra falsos positivos. Es posible llegar al punto de 100% de verdaderos positivos (en el 1 del eje de la ordenada), pero en ese punto la cantidad de falsos positivos es muy grande (se puede observar en las gráficas, hacia el sector arriba a la derecha).

El punto de operación adecuado se encuentra, entonces, en un punto en que la cantidad de verdaderos positivos reportada sea "alta", pero a la vez la cantidad de falsos positivos entregada sea adecuada para que tenga sentido económico para la empresa (justamente el problema que se pretende resolver, reducir las falsas alarmas).

La forma de la curva ROC obtenida para el modelo PART muestra que con este modelo es posible obtener un mejor resultado en cuanto a balance (hay puntos en la curva de mayor valor para verdaderos positivos y menor valor de falsos positivos) que en la curva para el modelo SVM. Es decir, es posible alcanzar un punto de operación más adecuado, con una mayor cantidad de verdaderos positivos y menor cantidad de falsos positivos, en el rango mostrado en la Figura 62.

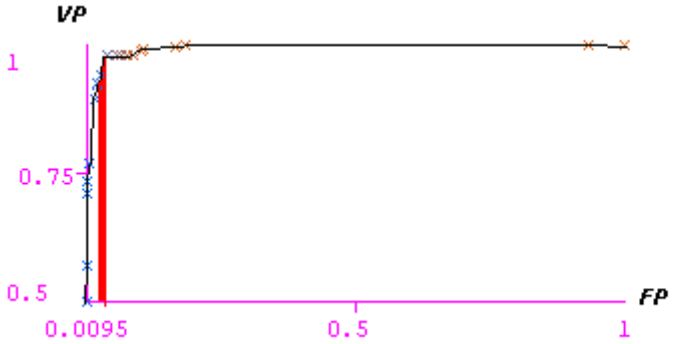


Figura 62: Rango del punto de operación

Por otra parte, se realizó una comparación práctica entre ambos

modelos.

Durante tres días de una semana cualquiera, se corrieron ambos modelos paralelamente con datos reales, comparando sus resultados.

Se obtuvieron los siguientes resultados promedio:

- Similitudes reportadas por el modelo PART: 1037.
- Similitudes reportadas por el modelo SVM: 2029.
- Similitudes reales reportadas por el modelo PART: 664.
- Similitudes reales reportadas por el modelo SVM: 653.

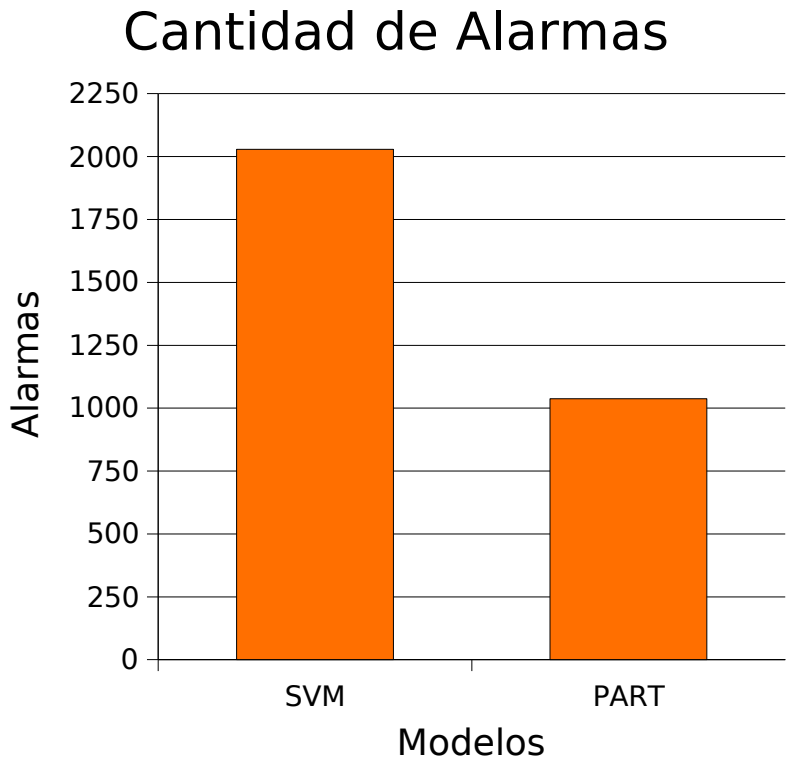


Figura 63: Alarmas arrojadas por ambos modelos

La comparación simple muestra que el modelo PART presenta, en promedio, el 50,6% de las alarmas presentadas por el modelo SVM.

Además, se aprecia que el modelo SVM presenta una menor cantidad de similitudes relevantes que el modelo PART (aunque una cantidad parecida: el 98%).

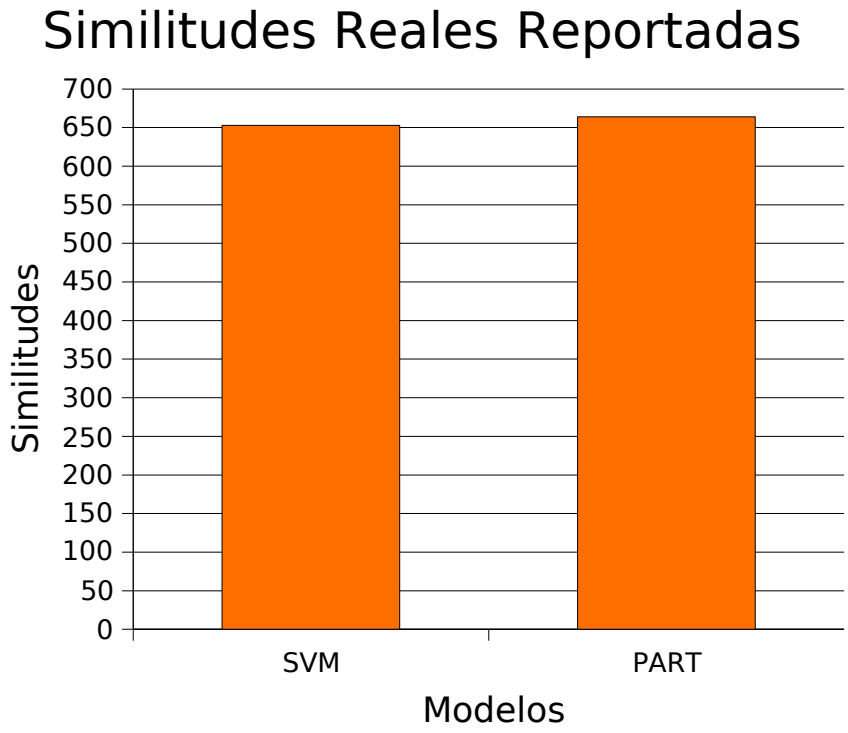


Figura 64: Similitudes reales entregadas por ambos modelos

Ante estos datos, si bien el análisis individual reporta que el modelo SVM entrega algunas similitudes que el modelo PART no presenta (6 en total, un 0,9% de los casos reportados por el modelo PART), el modelo SVM reporta más similitudes que no son entregadas por el modelo SVM (17 en total, un 2,6% de las similitudes entregadas por el modelo SVM).

Las falsas alarmas del modelo PART son, entonces, 1037-664 = 384, y para el modelo SVM 2029-653 = 1376.

El modelo SVM presenta un un 67,8% de falsas alarmas, mientras el modelo PART solamente un 37%.

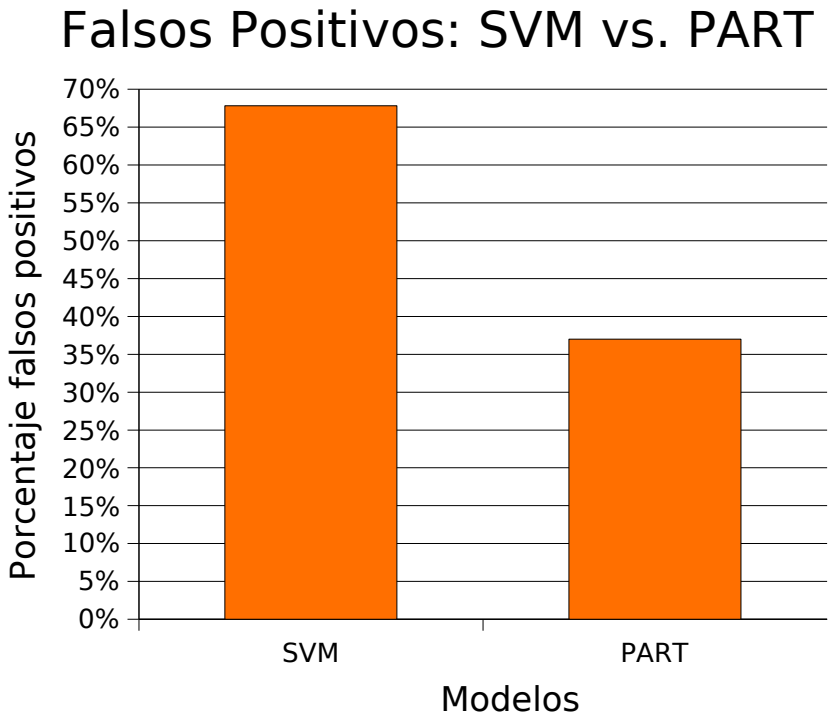


Figura 65: Falsos positivos o "falsas alarmas" reportadas por ambos modelos.

Si bien los números resultantes de la validación de los modelos pudo haber dejado alguna duda de la conveniencia de uno u otro modelo, dado lo estrecho de las diferencias, el resultado de esta prueba es implacable: el modelo PART presenta, en la práctica, resultados mucho más convenientes, reduciendo considerablemente las falsas alarmas.

Todos los siguientes análisis se realizan, entonces, considerando el modelo PART, que es el escogido definitivamente para realizar el trabajo de clasificación.

5.4 Pruebas diarias.

Una vez escogido el modelo, se procedió a probarlo durante un período de tres semanas. De este período de tiempo de prueba, se escogieron siete días de forma aleatoria, no consecutivos, para analizar y comparar resultados.

El procedimiento es el siguiente:

- Se considera, para cada día a estudiar, la cantidad de posibles similitudes reportadas por el modelo PART.
- Para los mismos días, se considera la cantidad de posibles similitudes reportadas por el algoritmo antiguo.
- Se calcula la proporción de alarmas entregadas por ambos algoritmos.
- Se obtiene la cantidad de similitudes reportadas por el algoritmo antiguo no reportadas por el sistema nuevo.
- Se considera la cantidad de similitudes reportadas por el sistema nuevo no reportadas por el sistema antiguo.
- Se calcula un "total de similitudes reportadas", considerando las similitudes reales entregadas por ambos algoritmos (coincidentes y no coincidentes). Sobre este total, se realiza una comparación porcentual de las similitudes entregadas por ambos modelos.

5.4.1 Primera prueba.

El resumen de casos se presenta a continuación:

ALGORITMO NUEVO	
casos reportados	1036
verdaderos positivos	662
falsos positivos	374
porcentaje verdaderos positivos	63.90%
porcentaje falsos positivos	36.10%
ALGORITMO ANTIGUO	
casos reportados	3047
verdaderos positivos	664
falsos positivos	2383
porcentaje verdaderos positivos	21.79%
porcentaje falsos positivos	78.21%
Similitudes totales reportadas:	758
similitudes encontradas por el algoritmo nuevo:	87.34%
similitudes encontradas por el algoritmo antiguo:	87.60%

Tabla 34: Resumen de casos de la primera prueba con datos "reales".

Se puede apreciar en el estudio que el algoritmo nuevo detecta algunas similitudes menos que el algoritmo antiguo, pero es bastante más eficiente en cuanto a la cantidad de falsos positivos entregados.

En el caso del algoritmo nuevo, el 36,10% de los casos corresponden a falsos positivos, mientras en el caso del algoritmo antiguo un 78,21% de los casos corresponden a este tipo de error, entregando, de este modo, un número muy abultado de falsas alarmas.

Por otra parte, ambos algoritmos entregan alarmas que el otro no es capaz de detectar. Los porcentajes de similitudes entregadas por ambos algoritmos son similares. Es decir, omiten cantidades similares de similitudes.

El algoritmo nuevo, sin embargo, es perfectible en este sentido, mediante la mejora de la tabla "diccionario", lo cual hace más eficiente el

cálculo de la característica "penalización".

Cada uno de los días de prueba se observan los casos que el algoritmo nuevo omitió, se analiza qué cadenas de caracteres puede ser útil agregar al diccionario, y la prueba siguiente se hace con un diccionario mejorado en comparación con el del día anterior.

5.4.2 Segunda prueba.

El resumen de casos se presenta a continuación:

ALGORITMO NUEVO	
casos reportados	1234
verdaderos positivos	707
falsos positivos	527
porcentaje verdaderos positivos	57.29%
porcentaje falsos positivos	42.71%
ALGORITMO ANTIGUO	
casos reportados	3739
verdaderos positivos	715
falsos positivos	3024
porcentaje verdaderos positivos	19.12%
porcentaje falsos positivos	80.88%
Similitudes totales reportadas:	813
similitudes encontradas por el algoritmo nuevo:	86.96%
similitudes encontradas por el algoritmo antiguo:	87.95%

Tabla 35: Resumen de casos de la segunda prueba con datos "reales".

Se aprecia en esta prueba también una mayor eficiencia en cuanto al número de falsas alarmas entregadas por parte del algoritmo propuesto como solución.

Si bien hay cerca del 1% de diferencia a favor del algoritmo antiguo en relación a las similitudes encontradas sobre el total de similitudes encontradas por ambos algoritmos, hay cerca del 40% de diferencia a favor del algoritmo nuevo en cuanto a éxito de los reportes entregados.

5.4.3 Tercera prueba.

El resumen de casos se presenta a continuación:

ALGORITMO NUEVO	
casos reportados	1384
verdaderos positivos	885
falsos positivos	499
porcentaje verdaderos positivos	63.95%
porcentaje falsos positivos	36.05%
ALGORITMO ANTIGUO	
casos reportados	4325
verdaderos positivos	887
falsos positivos	3438
porcentaje verdaderos positivos	20.51%
porcentaje falsos positivos	79.49%
Similitudes totales reportadas:	990
similitudes encontradas por el algoritmo nuevo:	89.39%
similitudes encontradas por el algoritmo antiguo:	89.60%

Tabla 36: Resumen de casos de la tercera prueba con datos "reales".

Se confirma la tendencia vista hasta el momento. El algoritmo antiguo, en esta tercera prueba, sigue encontrando un número levemente superior de similitudes que el algoritmo nuevo.

De todos modos, la diferencia es más estrecha que en la prueba anterior, reduciéndose a un 0,21%.

La diferencia en cuanto a porcentaje de verdaderos positivos sobre todos los positivos reportados por cada algoritmo, en tanto, se mantiene por sobre el 40% a favor del algoritmo nuevo.

5.4.4 Cuarta prueba.

El resumen de casos se presenta a continuación:

ALGORITMO NUEVO	
casos reportados	998
verdaderos positivos	638
falsos positivos	360
porcentaje verdaderos positivos	63.93%
porcentaje falsos positivos	36.07%
ALGORITMO ANTIGUO	
casos reportados	3110
verdaderos positivos	635
falsos positivos	2475
porcentaje verdaderos positivos	20.42%
porcentaje falsos positivos	79.58%
Similitudes totales reportadas:	711
similitudes encontradas por el algoritmo nuevo:	89.73%
similitudes encontradas por el algoritmo antiguo:	89.31%

Tabla 37: Resumen de casos de la cuarta prueba con datos "reales".

En esta oportunidad, el algoritmo nuevo reportó un número levemente superior de similitudes reportadas sobre el total de similitudes encontradas por ambos algoritmos. La diferencia es estrecha, de todos modos.

En efecto, la diferencia es ahora a favor del algoritmo nuevo, pero no superior a un 0,43%. Además de los números, el análisis caso a caso revela que el sistema nuevo, al irse completando el diccionario, entrega similitudes que en días anteriores posiblemente no habría entregado. Es decir, es posible sugerir que el sistema ha mejorado su rendimiento a lo largo del tiempo de prueba.

El porcentaje de verdaderos positivos sobre la cantidad total total de positivos entregados por cada sistema sigue favoreciendo al sistema nuevo por más del 40%.

Los falsos positivos, como se aprecia, son la tabla, disminuyen bastante

del algoritmo antiguo al nuevo. En efecto, el algoritmo nuevo presenta el 14,54% de falsos positivos en relación al algoritmo antiguo. Esta es, naturalemente, la principal fortaleza de la solución propuesta, que, al reducir los falsos positivos de este modo, logra su objetivo de reducir drásticamente la cantidad de "falsas alarmas" reportadas.

5.4.5 Quinta prueba.

El resumen de casos se presenta a continuación:

ALGORITMO NUEVO	
casos reportados	1013
verdaderos positivos	672
falsos positivos	341
porcentaje verdaderos positivos	66.34%
porcentaje falsos positivos	33.66%
ALGORITMO ANTIGUO	
casos reportados	3078
verdaderos positivos	659
falsos positivos	2419
porcentaje verdaderos positivos	21.41%
porcentaje falsos positivos	78.59%
Similitudes totales reportadas:	748
similitudes encontradas por el algoritmo nuevo:	89.84%
similitudes encontradas por el algoritmo antiguo:	88.10%

Tabla 38: Resumen de casos de la quinta prueba con datos "reales".

Nuevamente en este caso el algorirmo nuevo reporta una cantidad superior de similitudes encontradas que el algoritmo antiguo. La diferencia porcentual es levemente mayor que en el caso anterior (1,74%).

Por otra parte, aunque aumenta el número de similitudes encontradas, no se aprecia un aumento en el porcentaje de falsos positivos.

La relación verdaderos positivos / positivos reportados sigue favoreciendo, en efecto, al algoritmo nuevo en más de un 40%.

En cuanto a falsos positivos, el algoritmo nuevo entrega el 14,1% de los entregados por el sistema antiguo.

5.4.6 Sexta prueba.

El resumen de casos se presenta a continuación:

ALGORITMO NUEVO	
casos reportados	889
verdaderos positivos	581
falsos positivos	308
porcentaje verdaderos positivos	65.35%
porcentaje falsos positivos	34.65%
ALGORITMO ANTIGUO	
casos reportados	2693
verdaderos positivos	572
falsos positivos	2121
porcentaje verdaderos positivos	21.24%
porcentaje falsos positivos	78.76%
Similitudes totales reportadas:	652
similitudes encontradas por el algoritmo nuevo:	89.11%
similitudes encontradas por el algoritmo antiguo:	87.73%

Tabla 39: Resumen de casos de la sexta prueba con datos "reales".

Se mantiene la tendencia de los resultados de las dos pruebas anteriores: el algoritmo nuevo reporta un mayor número de similitudes que el algoritmo antiguo.

El porcentaje de falsos positivos se mantiene dentro del mismo orden de magnitud. En efecto, es levemente inferior que en la prueba anterior.

En efecto, se puede apreciar en el cuadro resumen que la relación falsos positivos / positivos totales se mantiene, para ambos algoritmos, en valores similares a los resultados de las pruebas anteriores.

La cantidad de falsos positivos reportados por el algoritmo nuevo corresponden al 14,5% de los reportados por el sistema antiguo.

5.4.7 Séptima prueba.

El resumen de casos se presenta a continuación:

ALGORITMO NUEVO	
casos reportados	1114
verdaderos positivos	711
falsos positivos	403
porcentaje verdaderos positivos	63.82%
porcentaje falsos positivos	36.18%
ALGORITMO ANTIGUO	
casos reportados	3481
verdaderos positivos	702
falsos positivos	2779
porcentaje verdaderos positivos	20.17%
porcentaje falsos positivos	79.83%
Similitudes totales reportadas:	798
similitudes encontradas por el algoritmo nuevo:	89.10%
similitudes encontradas por el algoritmo antiguo:	87.97%

Tabla 40: Resumen de casos de la séptima prueba con datos "reales".

En esta última prueba, el algoritmo nuevo presenta una vez más un porcentaje mejor de similitudes encontradas que el algoritmo antiguo, con una diferencia algo superior que en las pruebas anteriores.

La relación falsos positivos / positivos totales se mantiene en los mismos rangos que en las otras pruebas, tanto para el algoritmo nuevo como para el antiguo y, por lo tanto, la diferencia entre ambos también se mantiene dentro de los rangos apreciados en las pruebas anteriores.

Nuevamente el algoritmo nuevo entrega una cantidad cercana al 14,5% de falsos positivos que el sistema antiguo.

5.4.8 Resumen de pruebas.

El promedio de los resultados obtenidos en las pruebas se presenta en la siguiente tabla:

ALGORITMO NUEVO	
casos reportados	1095.43
verdaderos positivos	693.71
falsos positivos	401.71
porcentaje verdaderos positivos	63.51%
porcentaje falsos positivos	36.49%
ALGORITMO ANTIGUO	
casos reportados	3353.29
verdaderos positivos	690.57
falsos positivos	2662.71
porcentaje verdaderos positivos	20.67%
porcentaje falsos positivos	79.33%
Similitudes totales reportadas:	781.43
similitudes encontradas por el algoritmo nuevo:	88.78%
similitudes encontradas por el algoritmo antiguo:	88.32%

Tabla 41: Tabla resumen de los resultados de las pruebas: promedios.

A continuación, la visualización gráfica de las similitudes encontradas por uno y otro algoritmo, como porcentaje de todas las similitudes encontradas por ambos en conjunto:

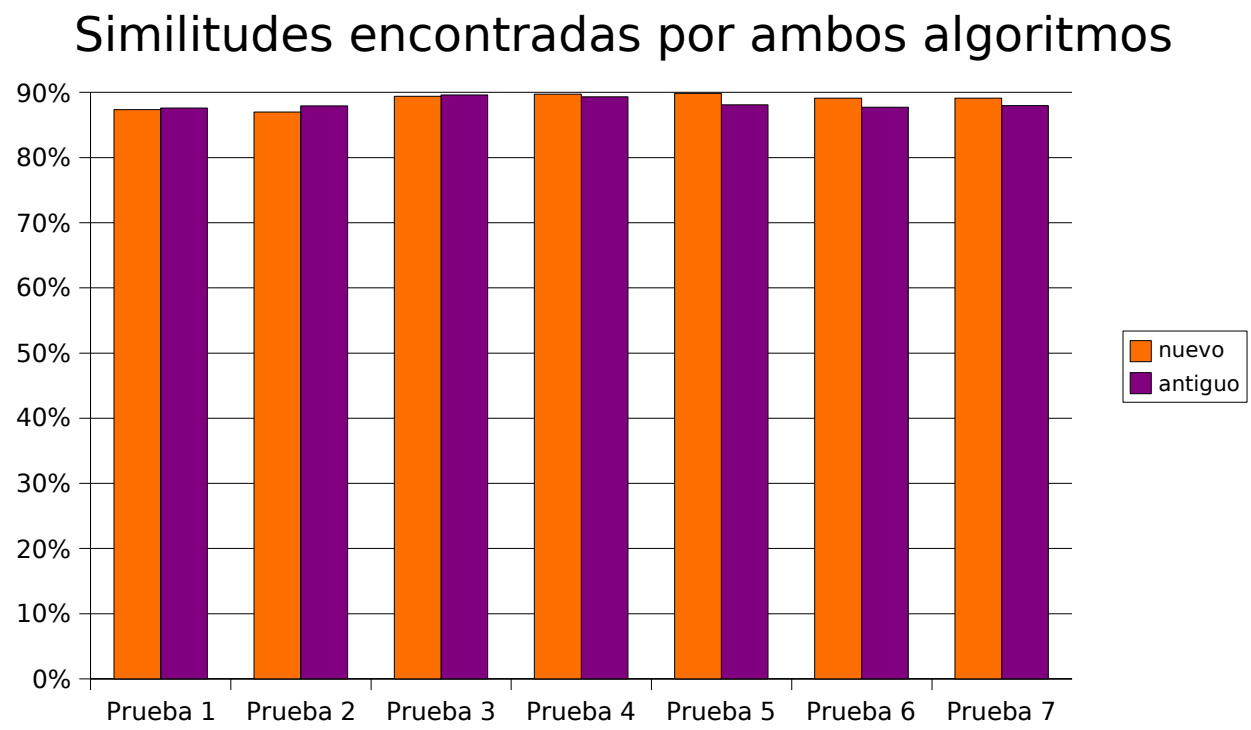


Figura 66: Similitudes encontradas por ambos algoritmos a lo largo de las siete pruebas presentadas.

Se puede apreciar de forma clara en la gráfica que ambos algoritmos presentan cantidades similares de similitudes a lo largo de las pruebas. También se puede apreciar que el algoritmo nuevo entrega un número mayor de similitudes a partir de la cuarta prueba.

Por otra parte, se presenta a continuación una gráfica comparativa del porcentaje de éxito de ambos algoritmos, medido como porcentaje de verdaderos positivos sobre la cantidad de falsos más verdaderos positivos encontrados a lo largo de las siete ejecuciones presentadas:

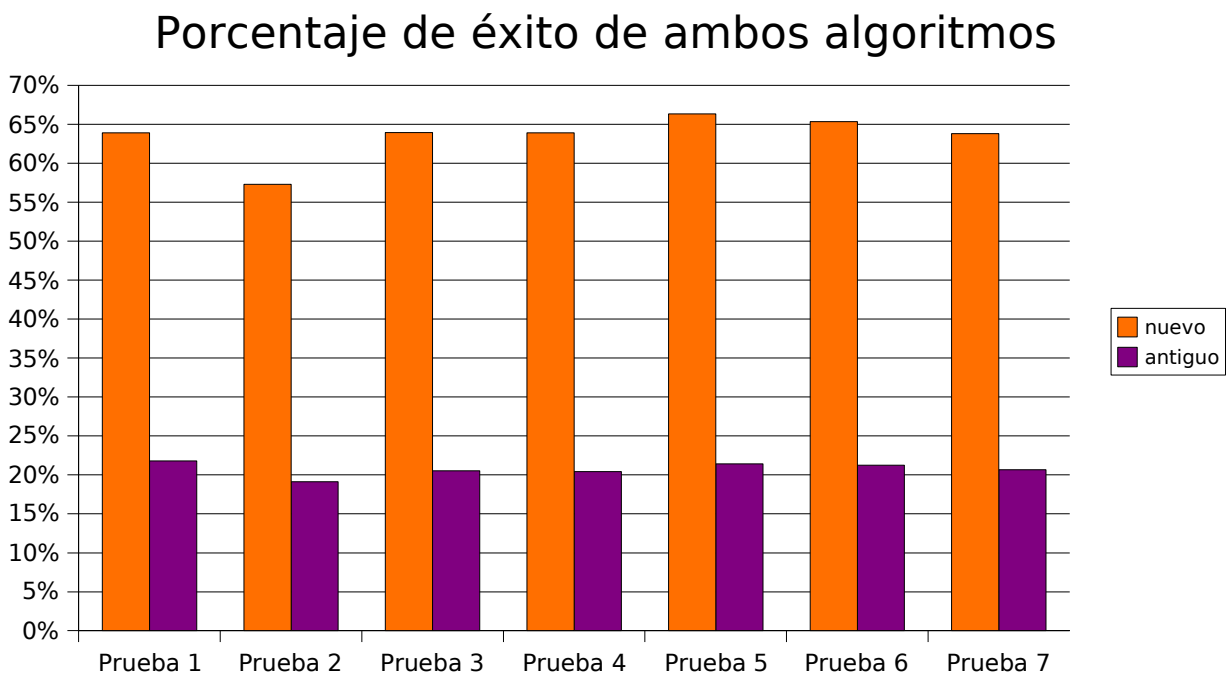


Figura 67: Porcentaje de éxito de ambos algoritmos, a lo largo de las siete pruebas presentadas.

Se aprecia en esta gráfica que el porcentaje de éxito del algoritmo nuevo es en todas las ejecuciones muy superior al del algoritmo antiguo, aproximadamente triplicándolo cada vez.

Es decir, de las alarmas reportadas por el algoritmo nuevo, un porcentaje mucho mayor corresponde a alarmas relevantes que en el caso del algoritmo antiguo, aproximadamente triplicando su efectividad.

Es decir, con todo lo anterior, se puede apreciar que el algoritmo nuevo mejora considerablemente la eficacia en el reporte de alarmas reales, disminuyendo las falsas alarmas reportadas y omitiendo un número similar de alarmas irrelevantes.

En efecto, del cuadro resumen presentado (Tabla 41), se puede extraer que el promedio de alarmas entregadas por el algoritmo nuevo asciende a 1095,43. Por otra parte, el promedio de alarmas entregadas por el algoritmo

antiguo llega a 3353,29. Es decir, en promedio el algoritmo nuevo impone la tarea a los supervisores humanos de revisar un 32,67% de los casos que obliga a revisar el algoritmo antiguo.

De esta cantidad de alarmas reportadas a diario, se puede también extraer de la tabla que en el algoritmo nuevo, en promedio, 401,71 corresponden a falsos positivos. En el algoritmo antiguo son, en promedio, 2662,71. Es decir, la cantidad de falsos positivos se reduce al 15,08%.

Suponiendo que el tiempo de revisión de las similitudes es lineal con la cantidad de alarmas entregadas, se obtiene que las personas encargadas de tal labor disminuyen el tiempo de revisión desde un promedio cercano a las seis horas hasta uno cercano a una hora con cincuenta y siete minutos.

Tal disminución de tiempo implica para la empresa un importante ahorro, equivalente a tener tres personas más contratadas durante dos tercios de la jornada, sin tener que pagar más sueldos, con el beneficio económico que esto pueda traer a la empresa.

6 Discusión de resultados y conclusiones.

En primer lugar, es importante analizar el trabajo presentado en relación a los objetivos planteados inicialmente.

Los números que se muestran en este informe revelan que se logró el objetivo de diseñar e implementar un algoritmo que mejora la eficiencia del sistema previo que la empresa contratante utilizaba para la detección de similitudes.

Se logró esto, además, cumpliendo los puntos que se planteaban en el capítulo de objetivos específicos: se observó y se describió el método utilizado por la empresa contratante. Se modeló el problema, convirtiendo el planteamiento inicial en un problema numérico equivalente clasificable mediante algoritmos de aprendizaje como los probados. Se realizó un trabajo de selección, diseño e integración de algoritmos y técnicas. El funcionamiento de la solución implementada se corresponde en un 100% con el diseño, lo cual habla de una implementación exitosa. Se realizó selección de datos para entrenamiento, validación y pruebas. Finalmente, fue posible probar el algoritmo y compararlo con su antecesor utilizando grandes volumenes de datos “reales”, es decir, obtenidos de la aplicación diaria.

Mientras el algoritmo antiguo presentaba siempre un número de falsos positivos mayor que el de verdaderos negativos (falsos positivos : 79,33% verdaderos positivos: 20,67%, según promedio de las pruebas), el nuevo algoritmo diseñado presenta un número minoritario de falsos positivos (36,49% contra 63,51%), lo cual es un signo de mayor eficiencia en este sentido.

En efecto, si el objetivo planteado era reducir a la mitad la cantidad de alarmas entregadas diariamente, se logró diseñar un sistema que reduce este resultado a aproximadamente un tercio (exactamente el 32,67%, según el promedio de las pruebas realizadas).

Parece claro, entonces, que el algoritmo nuevo consigue el objetivo de reducir las falsas alarmas, superando incluso la meta impuesta. La pregunta es: ¿logra el objetivo de realizar esto sin aumentar significativamente las omisiones?

Es decir, ¿logra el objetivo de ser más eficiente en el control de errores de tipo I, pero sin hacerse infactible debido al aumento de errores de tipo II?

La inspección de los resultados de las pruebas revela que los errores de tipo II no presentan aumento.

De todos modos, es necesario precisar que no se sabe el número exacto de errores de tipo II, ya que se tiene información solamente de las alarmas entregadas, no de aquellas omitidas.

Precisando lo anterior, se tiene información sobre las alarmas entregadas por ambos sistemas y, por tanto, sobre las alarmas entregadas por un algoritmo y omitidas por el otro y vice-versa.

No se dispone de las alarmas omitidas por ambos sistemas. Por lo tanto, la información que se puede extraer sobre los errores de tipo II solamente es útil en términos comparativos entre ambos algoritmos, no como cuantificación total.

Tomando lo anterior en cuenta, se puede apreciar en los resultados presentados que ambos algoritmos detectan algunos resultados que el otro no es capaz de detectar.

Sin embargo, en cuanto a números, ambos algoritmos detectan cantidades similares de similitudes. En las primeras pruebas, se podía apreciar que el algoritmo antiguo presentaba una cantidad levemente superior de similitudes que el algoritmo nuevo.

En las últimas, en cambio, este resultado se invirtió, teniéndose que el algoritmo nuevo entrega un número levemente superior de similitudes. Este cambio se puede adjudicar a que en el transcurso de la etapa de pruebas se

fue completando la tabla "diccionario", que inicialmente era bastante básica. En las pruebas iniciales, se tenía una tabla diccionario con 104 términos. Hacia el final de la etapa de pruebas, solamente pensando en casos nuevos y observando los resultados de las pruebas (especialmente errores de tipo II del nuevo algoritmo), el número de entradas del diccionario aumentó a 902.

Es necesario comentar que la tasa de ingresos en la tabla diccionario disminuyó considerablemente en la medida en que se realizaban pruebas. Es decir, si luego del primer día de pruebas el diccionario aumentó en un número cercano a las 200 entradas, el último día de pruebas el crecimiento de la tabla no superó las 10 entradas, lo cual coincide con que el sistema cometía un menor número de errores de tipo II en comparación con el algoritmo antiguo.

En este sentido, se cumple la otra condición impuesta, es decir, el sistema nuevo no "dispara" la cantidad de falsos negativos, a pesar de disminuir la cantidad de falsos positivos.

Además, en el resumen de resultados se puede apreciar que, a pesar de aumentar el tamaño de la tabla diccionario, no aumenta el número de falsos positivos. Esto confirma que, al aumentarse el número de entradas del diccionario de un modo coherente, aumenta el número de similitudes encontradas, pero no el número de falsas alarmas.

La alimentación de esta tabla diccionario es relevante, por el peso que toma durante el proceso de entrenamiento la característica "penalización".

Uno de los problemas de la solución propuesta está precisamente relacionado con esta característica y su diccionario asociado. En efecto, es imposible construir una tabla que contenga todas las posibles secuencias de caracteres de interés.

Sin embargo, la inspección de casos revela que en su mayoría, las secuencias de caracteres "acompañantes" se repiten, y aunque no se puede consturir un diccionario con todas las posibilidades, si se puede construir un diccionario lo suficientemente amplio para tener un margen de error razonable.

En efecto, los resultados de las pruebas revelan que el algoritmo nuevo presenta omisiones, pero el margen de error es razonable dentro de los parámetros impuestos (comparación con el algoritmo antiguo), y la cantidad de errores de tipo II se reduce considerablemente.

Por lo tanto, se puede afirmar que los objetivos del trabajo fueron cumplidos. Además, esto se logró mediante una solución original, que no se copió de ninguna publicación ni aplicación existente, diseñada especialmente para el problema propuesto.

En algunos sentidos, como la elección del modelo PART por sobre el SVM y MLP, el criterio seguido en la implementación privilegia la disminución de las falsas alarmas por sobre las omisiones. Algunos de los modelos generados prometían, al menos en el proceso de validación, un número más reducido de omisiones, a costa de un número muy superior de falsas alarmas.

Posiblemente la aplicación de algunos de estos modelos habría entregado un mayor éxito en cuanto a la reducción de omisiones, logrando resultados bastante superiores a los conseguidos con el algoritmo nuevo y con el algoritmo antiguo, y aún presentar un número menor de falsas alarmas de las conseguidas con el algoritmo antiguo.

Es muy importante señalar que todo lo que se ha expuesto al momento está pensado sobre la base de los resultados obtenidos. Es decir, se habla de los modelos SVM, MLP y PART obtenidos exclusivamente en este trabajo.

Toda una nueva rama de discusión se abre al considerar que con un proceso de entrenamiento distinto y con diferencias metodológicas (por ejemplo, una construcción diferente del conjunto de entrenamiento, que pudo haber sido balanceado 50% similitudes y 50% no similitudes, en lugar del 48& - 52% utilizado), se podrían haber obtenidos diferentes resultados.

Es, por lo tanto, en extremo importante decir que cuando se concluye sobre los resultados obtenidos con PART, MLP y SVM solamente se hace referencia a los modelos obtenidos en este proceso, y no se extiende a la

generalidad.

El foco principal del trabajo consistía esencialmente en reducir las falsas alarmas, sin aumentar las omisiones, no a reducir dramáticamente estas últimas. Y, considerando que con un mejor diccionario de todos modos se puede reducir la cantidad de omisiones, se escogió el modelo utilizado finalmente en la solución, y se estima que resultó finalmente una buena elección.

Cabe preguntarse, de todos modos, por qué, si bien los resultados en la validación de los tres tipos de modelos son parecidos, el modelo PART utilizado presenta resultados tan superiores al otro modelo probado (SVM).

Existen varios factores a considerar. Uno de ellos, es la construcción de los conjuntos de entrenamiento y validación. Es posible que los conjuntos construidos se hayan inclinado por ciertos tipos de patrones, y que la aleatoriedad de los casos obtenidos a diario acentúe profundamente las diferencias.

Por otra parte, al considerar el universo total de casos a clasificar diariamente (todos los posibles pares nueva solicitud – dominio protegido), se habla, cada día, de un universo de millones de instancias a revisar. Sobre un universo de tal tamaño, una pequeña diferencia porcentual es, a la vez, una diferencia en un gran *número* de casos. Hay que subrayar, como se ha mencionado anteriormente, que la mayoría de los casos de este universo son descartados por los tres modelos, y no se posee información sobre ellos.

Es decir, probablemente las diferencias porcentuales sobre el total del universo de casos no es tan grande, aunque la diferencia, en un sentido estrictamente práctico, sea lo suficientemente grande para que la empresa que solicita el trabajo prefiera una solución por sobre la otra.

Al terminar el proceso de diseño, se hace la recomendación a la empresa contratante de seguir mejorando el diccionario: en efecto, en la empresa hay personal con conocimiento suficiente para realizar esta tarea, de mejor forma

que quien diseñó el algoritmo.

Se proponen algunos pasos en este sentido:

- Pensar en términos adecuados que se hayan presentado históricamente e incluirlos en el diccionario. Estos términos no deben ser palabras de menos de tres caracteres, pues estos aumentan la posibilidad de obtener falsas alarmas.
- Tener las salidas del nuevo algoritmo en observación durante un período de tiempo (estimado en uno o dos meses), comparar los resultados con los del algoritmo antiguo y en base a esto seguir completando el diccionario.

Estos pasos no son imprescindibles, y de hecho, como se mostró, los resultados ya cumplen con el objetivo impuesto, y mejoran los resultados obtenidos que la empresa conseguía con el algoritmo antiguo. Si se sigue el consejo, sin embargo, el sistema puede eventualmente presentar resultados incluso mejores. La decisión de seguir estos pasos o no depende exclusivamente de la empresa contratante, en caso de justificarse el costo de realizar esta operación, considerando que el objetivo por ellos perseguido ya está cumplido con el trabajo entregado.

Un contratiempo del nuevo algoritmo radica en el hecho de que las reglas seguidas para la clasificación no están programadas, sino que se aprenden del proceso de entrenamiento, lo cual es un principio totalmente distinto del algoritmo antiguo.

Dado que en el algoritmo antiguo estaba todo programado y las reglas eran todas conocidas de antemano, y discutidas entre el programador y otros integrantes de la empresa, las limitaciones del sistema eran claramente conocidas de antemano. A esto se debe sumar que las reglas del algoritmo antiguo eran muy simples y se basaban en la comparación directa de caracteres.

De este modo, la empresa tenía la opción de advertir a sus clientes de

estas limitaciones previamente a entregar el servicio.

Sin embargo, con la nueva solución esto no es tan claro. Las reglas son más complejas, y las decisiones se toman según combinaciones no simples de estas reglas. Por lo tanto, no es tan simple advertir a los clientes sobre las posibles limitaciones del sistema.

Sin embargo, a cambio de esto, el nuevo sistema supone ventajas en cuanto a ahorro de tiempo y dinero (lo cual se intentó cuantificar en el capítulo anterior), y posiblemente una mayor efectividad en cuanto a omisiones, en la medida en que se mejore la construcción del diccionario.

Además, el uso de un algoritmo "inteligente" para la protección de dominios abre nuevas posibilidades en cuanto a marketing al ofrecer el servicio a nuevos potenciales clientes, lo cual también fue discutido con el contratante durante el tiempo de trabajo.

Posiblemente se pudo haber ideado alguna estrategia para conseguir mejores resultados, como las posibilidades que se han planteado en estas líneas (utilizar otro de los modelos de clasificador entrenados, por ejemplo), o la anidación de clasificadores, por ejemplo, aplicando otro clasificador a la salida del primero.

No se hizo, sin embargo, porque al conseguir los resultados esperados se desestimó la necesidad de realizar un trabajo más profundo en este sentido.

Sin embargo, aunque se consideró suficiente el resultado del trabajo al cumplir los objetivos propuestos (incluso superados), esto abre posibilidades a trabajos futuros en la materia, para lograr reducir aún más los errores y lograr algoritmos más efectivos para la solución de este tipo de problemáticas.

Otras posibles líneas de trabajo consisten en la adaptación del algoritmo propuesto y las posibles mejoras sobre él a problemas de mayor escala y de importancia más crítica, como es la protección de marcas comerciales, en que posiblemente la reducción de error de tipo II (falsos negativos) es de mayor relevancia que en el problema que se trató en este trabajo.

7 Referencias.

1. BERGER, Helmut / MERKL, Dieter / DITTENBACH, Michael: *Exploiting Partial Decision Trees for Feature Subset Selection in e-Mail Categorization.* 2006. ACM Special Interest Group on Applied Computing.
2. CNN Online: *Microsoft takes on teen's site MikeRoweSoft.com*, http://www.cnn.com/2004/TECH/internet/01/19/offbeat.mike.rowe.soft.ap/
3. ESTEVEZ, Pablo: *Apuntes del curso de Redes Neuronales.*Universidad de Chile.
4. FAWCETT, Tom: *ROC Graphs: Notes and Practical Considerations for Data Mining Researchers*. 2003. HP Laboratories.
5. FRAKES, William: *Stemming Algorithms.1992.* Prentice-Hall, Inc.
6. OFLAZER, Kemal / GÜZEY, Cemaleddin: *Spelling Corection in Agglutinative Languages.* 1994. Proceedings of the Fourth ACL Conference on Applied Natural Language Processing.
7. SAHA, Angshuman: *Introduction to Artificial Neural Network Models.* http://www.geocities.com/adotsaha/
8. SAHINALP, Cenk / TASAN, Murat / MACKER, Jai / OZSOYOGLU, Meral: *Distance Based Indexing for String Proximity Search.* 2003. ICDE.
9. SWENSON, Krister / MARRON, Mark / EARNEST-DE YOUNG, Joel / MORET, Bernard: *Approximating the True Evolutionary Distance Between Two Genomes*. 2004. University of New Mexico.
10. UNIVERSITY OF WAIKATO: *Weka Machine Learning Project – API*. http://weka.sourceforge.net/doc
11. UNIVERSITY OF WAIKATO: *The Weka Wiki*. http://weka.sourceforge.net/wiki/

12.YANCEY, William: *An Adaptive String Comparator for Record Linkage.* 2004. U.S. Census Bureau, Statistical Research Division.

Anexos.

A) Distancia de edición.

El problema de la distancia de edición consiste en encontrar la mínima cantidad de cambios para transformar un string en otro. Los tipos de cambio considerados son las siguientes operaciones:

- Inserción de un carácter.
- Sustitución de un carácter por otro.
- Eliminación de un carácter.

Existen varias posibles implementaciones de la solución a este problema. Una de ellas es recursiva:

Sea el string inicial *a,* de caracteres *a[1]...a[m]*, y el string de destino b, de caracteres *b[1]...b[n]*.

Sea la distancia d(i,j) la mínima cantidad de transformaciones (con las operaciones previamente definidas) para transformar *a[1]...a[i]* en *b[1]...b[j]*, es posible obtener d(i,0), es decir, la distancia entre *a[1]...a[i]* y el string vacío, y d(0,j).

A partir de lo anterior, es simple calcular de forma recursiva la distancia de edición como función de d(i,j-1), d(i-1,j) y d(i-1,j-1), del modo siguiente:

- Para i = 1 hasta m: d[i, 0] = i
- Para j = 1 hasta n: d[0, j] = j
- Para i =1 hasta m, j = 1 hasta n:
  - Si a[i] = a[j], costo = 0 (definición).
  - De lo contrario, costo = 1 (definición).
  - d[i, j] := mínimo(d[i-1, j] + 1, d[i, j-1] + 1, d[i-1, j-1] + costo), en que el primer parámetro representa eliminación, el segundo inserción y el tercero sustitución.
- Distancia de edición es d[m, n]

De este modo, bastante simple, es posible calcular la distancia de edición.

Sin embargo, el método recursivo puro no es eficiente: el costo en términos de programación es muy intensivo. En efecto, al calcular la distancia de edición entre strings de cierto tamaño, es empíricamente demostrable que el algoritmo se hace lento y es posible que en un pc normal existan problemas de memoria.

Por este motivo, en la práctica se utiliza tabulación para resolver este problema. Es decir, se almacenan resultados en una tabla, o en una matriz, para evitar que en el proceso recursivo se re-calculen los mismos resultados una y otra vez, calculando todo en una sola oportunidad, guardando en una matriz y, posteriormente, consultando los resultados guardados.

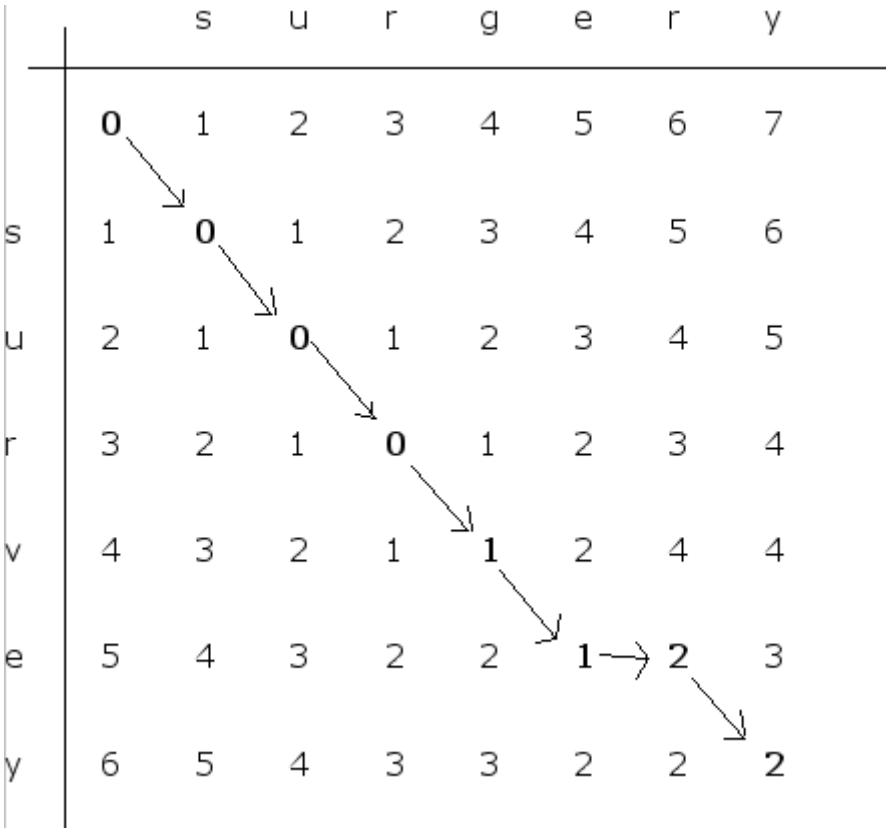


Figura 68: Matriz de cálculo de distancia de edición.

La matriz de costos presentada en la Figura 68 contiene, en la posición (i,j), el resultado de d(i,j). Al utilizar una matriz como esta, tal como está dicho, cada cálculo se hace una sola vez. Además, es sencillo visualizar gráficamente la función: cada cálculo de un mínimo representa un movimiento horizontal, vertical o diagonal en la matriz. Con esto, además es posible devolverse por los caminos de mínimo costo de la matriz y obtener estadísticas exactas de qué cambios es necesario realizar para transformar un string en el otro.

B) Estructura del código fuente.

El código fuente se escribió de forma modular, llegando, por facilidad de lectura y por orden, a tener varios archivos con las distintas secciones de código, en lugar de un solo archivo con un programa grande.

Así también, se dividió el programa en varias clases, aprovechando las facilidades de clases y objetos del lenguaje de programación.

La relación jerárquica entre las clases más importantes programadas se presenta a continuación:

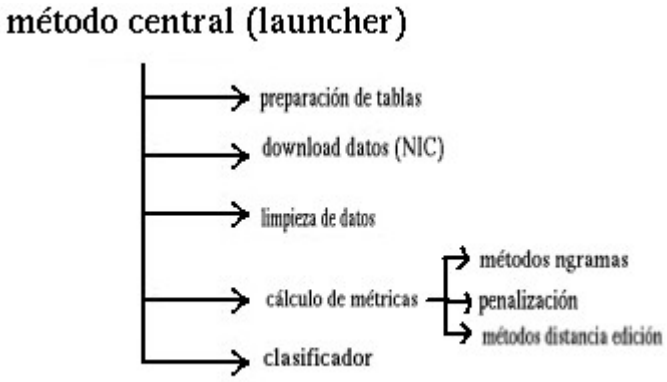


Figura 69: Jerarquía de clases programadas.

En los módulos mostrados en la Figura 69, el módulo central o "launcher" es el principal, que invoca a todos los otros (pero no hace nada por sí mismo).

El módulo "preparación de tablas" prepara las tablas de la base de datos para toda la operación posterior.

El módulo descrito como "download datos (NIC)" obtiene los datos desde el sitio web de NIC Chile.

El módulo "limpieza de datos" se encarga de extraer la información

relevante de los datos bajados desde el sitio de NIC.

"Cálculo de características" es, como su nombre indica, el módulo encargado de calcular las características, para lo cual se vale de otros módulos que implementan diversos métodos. Los módulos más importantes aparecen en la figura.

"Clasificador" es el módulo que se encarga de realizar la clasificación, es decir, aplica el modelo de clasificador generado previamente (PART, finalmente) para clasificar todos los datos en similitudes y no similitudes.