Universidad Central "Marta Abreu" de Las Villas

Facultad Matemática-Física-Computación

En opción del título:

Máster en Computación Aplicada

Tema:

Desarrollo de un predictor de interacciones entre dominios de proteínas basado en nuevos descriptores numéricos de secuencias de aminoácidos

Autor:

Ing. Sandra Romero Molina (sandra@uclv.cu)

Tutor:

Dr. Yasser B. Ruiz Blanco (yasserrb@uclv.edu.cu)

Dra: María Matilde García Lorenzo

Santa Clara, noviembre de 2016

DICTAMEN

Sandra Romero Molina, hago constar que el trabajo titulado “Desarrollo de un predictor de interacciones entre dominios de proteínas basado en nuevos descriptores numéricos de secuencias de aminoácidos” fue realizado en la Universidad Central “Marta Abreu” de Las Villas como parte del ejercicio para optar por el título de Máster en Computación Aplicada, autorizando a que el mismo sea utilizado por la institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos ni publicado sin la autorización de la Universidad.

_____

Firma del autor

Los abajo firmantes, certificamos que el presente trabajo ha sido realizado según acuerdos de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____ _____

Firma del tutor Firma del Jefe del Laboratorio

_____

Fecha

Agradecimientos

Agradezco a:

Mi familia, por ser la que me impulsa a que luche por ser alguien cada día mejor, por todas las horas en las que me cubrieron para que me dedicara solo a la tesis.

Mi tutor Yasser B. Ruiz Blanco, por haberme ayudado tanto a lo largo del trabajo, por ser tan competente y tan dedicado al trabajo.

Mi tutora María Matilde García Lorenzo, por todo su apoyo y colaboración.

A mi amigas Elizabeth Martínez Pérez y Maricel Meneses Gómez que me indicaron siempre el camino para hoy tener este logro.

A mis compañeros de trabajo y todas las personas que de una forma u otra me apoyaron siempre.

Sabemos lo que somos pero no lo que podemos llegar a ser

William Shakespeare

Resumen

Las proteínas intervienen en todos los procesos biológicos que ocurren con y entre las células. Particularmente las interacciones entre proteínas determinan muchas de sus funciones, como la acción enzimática en procesos metabólicos, la transmisión de señales y el transporte de sustancias. Por ello, el desarrollo de métodos computacionales de predicción de Interacciones Proteína-Proteína (PPI del inglés *Protein-Protein Interactions*) resulta uno de los objetivos más significativos de la Biología Computacional moderna. Los mejores métodos de predicción de PPI emplean el análisis de homología entre proteínas, mediante técnicas de alineamiento de secuencias de aminoácidos. Esta estrategia, unido al deficiente uso de casos de pares sin interacción, lleva a que los métodos muestren baja precisión, lo que limita su aplicabilidad para el diseño de proteínas/péptidos.

El presente trabajo describe los resultados obtenidos en la modelación de interacciones dominio-dominio (DDI) empleando nuevos descriptores numéricos libres de alineamiento calculados con el software ProtDCal. Se conforma una base de casos de pares de dominios de proteínas con interacción o ausencia de interacción; obtenidos de las bases de datos 3DID, iPFam y Negatome, en donde cada par de secuencias es representado por un conjunto de descriptores numéricos. Se realiza un pre-procesamiento del conjunto de entrenamiento y se construyen modelos empleando la técnica de aprendizaje automatizado *Máquinas de Vectores de Soporte*. Los resultados se comparan con métodos externos; mostrando un desempeño superior particularmente en la exactitud de las predicciones.

Abstract

Proteins are involved in all biological processes occurring within and between cells. Particularly protein interactions determine many of its functions such as enzyme action in metabolic processes, signal transmission and transport of substances. That is why the development of computational methods for predicting protein-protein interactions is one of the most important goals of modern Computational Biology. Best PPI prediction methods used to date analyze homology between proteins by amino acid sequences alignment techniques. This homology dependent strategy, coupled with poor use of case negative information leads to methods show low precision, which limits its applicability to design proteins / peptides.

This paper describes the results of the modeling domain-domain interactions (DDI) using new free alignment numerical descriptors calculated with ProtDCal software. An instances base is formed by pairs of protein domains with interaction or absence of interaction obtained from the 3DID, IPFam and Negatome databases; in which each pair of sequences is represented by a set of numerical descriptors. A pre-processing of the training set is performed and models are constructed using the machine learning technique *Support Vector Machines*. The results are compared with those shown by different external methods; showing superior performance particularly in the accuracy of predictions.

Índice de contenidos

Índice de tablas

Índice de figuras

Índice de ecuaciones

Introducción

Las proteínas intervienen en casi todos los procesos biológicos que ocurren con y entre las células; determinan su estructura y forma, y dirigen casi todos los procesos vitales (1). Sin embargo, raramente actúan solas, son las asociaciones complejas existentes entre ellas las que definirán el comportamiento final de un organismo. Tales interacciones determinan muchas de sus funciones, como la acción enzimática de numerosos procesos metabólicos, la transmisión de señales y el transporte de sustancias específicas dentro y fuera de las células (2). Comprender estas interacciones en detalle, proporciona información útil sobre el mecanismo molecular del funcionamiento de la célula, y ayuda en el diseño de fármacos más específicos y en la ingeniería de procesos celulares (3).

De ahí que el conocimiento en Biología Molecular haya transformado sucesivamente sus paradigmas a lo largo de los años, de la elucidación de Genomas, a Proteomas y más recientemente a Interactomas; siempre acercándose al objetivo final de conocer y diseñar el funcionamiento celular regido por las proteínas (4).

Algunas familias de proteínas son definidas por determinados fragmentos de la cadena de aminoácidos que son críticos para la función propia de la proteína (5). Estos fragmentos, a menudo se pliegan de forma autónoma respecto al resto de la proteína. En este sentido, aquellas regiones de una cadena polipeptídica que se pueden plegar de forma estable e independiente y a la vez dotar de características funcionales a las proteínas, son llamados dominios. A menudo, una de las funciones de los dominios es la interacción con otras proteínas. Las mayores bases de datos relacionadas con interacciones entre dominios son 3DID (6), iPFam (7) y Negatome (8), las cuales se encuentran disponibles en Internet para que la comunidad científica pueda hacer uso de las mismas en sus investigaciones. En ellas, se puede obtener información de forma rápida sobre pares de dominios de proteínas con conocimiento de interacción y sobre pares que hasta la fecha han mostrado muy baja probabilidad de hacerlo, obtenidos por métodos de experimentación científica.

Los métodos experimentales con los que se analiza la ocurrencia de interacción entre las proteínas son complejos, consumen mucho tiempo, y tienen un alto costo asociado a ellos (9). Como resultado el número de interacciones descubiertas hasta la fecha es limitado en comparación con el número total de pares de proteínas (posibles interacciones) que puede obtenerse dado el número de proteínas ya conocidas en la naturaleza (10). Lo mismo es aplicado en el caso de interacciones entre dominios, por ejemplo existen 16306 dominios conocidos almacenados en bases de datos como PFam (11), en principio la combinatoria a pares entre tal número de dominios genera ~$1.3x10^8$ posibles pares de dominios, aunque es esperado que muchos de estos pares no interaccionen realmente, resulta evidente el bajo número de interacciones reportadas, el cual es solo del orden de 10^3 combinando las entradas de las bases de datos 3DID e iPFam.

Todo lo anterior, ha conllevado a la creciente necesidad de desarrollar herramientas computacionales capaces de identificar tales interacciones (12). En consecuencia, el desarrollo de métodos computacionales de predicción de interacciones proteína-proteína resulta uno de los objetivos más significativos de la Biología Computacional moderna. Los mejores métodos de predicción de PPI hasta la fecha emplean el análisis de homología entre proteínas o fragmentos de las mismas, mediante técnicas de alineamiento de secuencias de aminoácidos, para desarrollar sus algoritmos de predicción (9). Sin embargo el uso limitado de casos de información negativa (no ocurrencia de interacción), lleva a que los métodos muestren baja precisión, lo que limita su aplicabilidad para el diseño de proteínas/péptidos con un patrón de PPI específico.

De lo anterior se deriva como **problema científico** del presente trabajo, la baja precisión de los métodos de predicción de PPI existentes en la actualidad.

Es posible identificar proteínas que presentan características "similares" basándose en codificaciones numéricas de las secuencias de aminoácidos (13). Una forma de representación de la información contenida en una secuencia de aminoácidos es mediante descriptores numéricos (14)(15)(16)(17); los cuales son el resultado final de un procedimiento lógico y matemático que transforma información química codificada dentro de una representación simbólica de una molécula en un número útil (18). En

consecuencia, el grupo de investigación de Descubrimiento de Fármacos y Bioinformática de la Facultad de Química-Farmacia desarrolló el programa ProtDCal (16), una herramienta para el cálculo de descriptores de proteínas con propósito general.

En la literatura es posible encontrar métodos contemporáneos que emplean descriptores numéricos de proteínas. Todos ellos, hacen uso de Máquinas de Vectores de Soporte (SVM del inglés *Support Vector Machines*) en la modelación de PPI (19)(20)(21)(22). Estos métodos constituyen las herramientas públicas de mayor acceso para predecir PPI a partir de secuencias de aminoácidos.

Se propone como **hipótesis** la siguiente**:** la introducción de nuevos descriptores de PPI mejora la precisión de la predicción de interacciones entre dominios de proteínas.

Por ello y dada la evidencia del uso de las SVM en este problema, el presente trabajo tiene como **objetivo general**: desarrollar modelos basados en Máquinas de Vectores de Soporte y nuevos descriptores numéricos de proteínas para la predicción de interacciones entre pares de dominios de proteínas.

El objetivo general se ha subdividido en los siguientes **objetivos específicos**:

1. Conformar un conjunto de entrenamiento, de alta fiabilidad, de pares de dominios con y sin interacción a partir de las mayores bases de datos existentes en la actualidad de ambos tipos de información.
2. Definir nuevos descriptores numéricos para pares de proteínas a partir de los descriptores introducidos en ProtDCal.
3. Seleccionar los atributos más representativos para el conjunto de entrenamiento.
4. Determinar la variante de SVM que mayor exactitud alcance empleando los atributos seleccionados.
5. Evaluar la propuesta con otros predictores internacionales.

6. Implementar un software para su futura aplicación y divulgación en la comunidad científica.

Aporte práctico

El aporte de la tesis es un software capaz de predecir interacciones entre dominios de proteínas con mayor precisión que otros predictores contemporáneos de PPI.

El presente informe de tesis consta de **tres capítulos** que se describen a continuación:

En el capítulo **"Marco Teórico"**, se abarcan los temas referentes al problema de investigación. Se describe la interacción entre dominios de proteínas y se realiza una revisión de las bases de datos donde se localizan dominios de proteínas con interacción y sin interacción reportada. Se introduce el concepto de descriptor numérico como una forma de representar numéricamente la información de las secuencias de aminoácidos de las proteínas y se describen los mejores métodos computacionales de predicción de PPI hasta la fecha.

En el capítulo **"Métodos a emplear en la solución del problema"** se recogen los métodos y técnicas a emplear para dar respuesta al problema planteado. Se describe la técnica de Aprendizaje Automático a emplear para la creación del clasificador: SVM. Se describen los métodos de selección de atributos a utilizar, así como los algoritmos que se podrían emplear para dividir los datos en entrenamiento y prueba. Por último, se describe cómo evaluar los modelos mediante medidas de evaluación, validación cruzada y la Curva ROC.

En el capítulo **"Sistema para predecir interacciones entre dominios de proteínas"** se detalla paso a paso el proceso de creación del clasificador, partiendo de la conformación de la base de casos y su división en entrenamiento y prueba. En este apartado se describen el proceso de selección de los atributos más relevantes para la modelación, el ajuste de los parámetros de SVM para alcanzar un modelo de alta precisión, la comparación del mejor modelo obtenido con los métodos computacionales contemporáneos más relevantes. Por último, se desarrolla un software para permitir su utilización por la comunidad científica.

Capítulo 1: Marco teórico

En el capítulo se describe la interacción entre dominios de proteínas. Se realiza una revisión de las principales bases de datos donde se localizan dominios de proteínas con interacción y sin interacción reportada. Se introducen los descriptores moleculares; introduciendo a ProtDCal, un programa para el cálculo de descriptores. Se describen los mejores métodos computacionales de predicción de PPI hasta la fecha.

1.1 ¿Qué son las proteínas?

Las proteínas pueden considerarse polímeros de unas pequeñas moléculas que reciben el nombre de aminoácidos. Los aminoácidos están unidos mediante enlaces peptídicos. La unión de un bajo número de aminoácidos da lugar a un péptido; si el número es superior a 50 aminoácidos se está en presencia ya de una proteína. Por consiguiente, las proteínas son cadenas de aminoácidos que se pliegan adquiriendo una estructura tridimensional que les permite llevar a cabo miles de funciones (23).

Químicamente, las proteínas son polímeros lineales formados por la unión del grupo carboxilo de un aminoácido con el grupo amino de otro aminoácido, a través de la formación de enlaces peptídicos covalentes. La formación de un dipéptido a partir de dos aminoácidos viene dado por la liberación de una molécula de agua (24). El péptido "columna vertebral" de una proteína consiste en la secuencia repetida -N-C_α-C-, donde la N representa el nitrógeno de amida, la C_α es el carbono en alfa de un aminoácido en la cadena del polímero, y la C final es el carbono del grupo carbonilo del aminoácido, que a su vez está vinculada a la amida N en el siguiente aminoácido en la línea (25) (Ver figura 1).

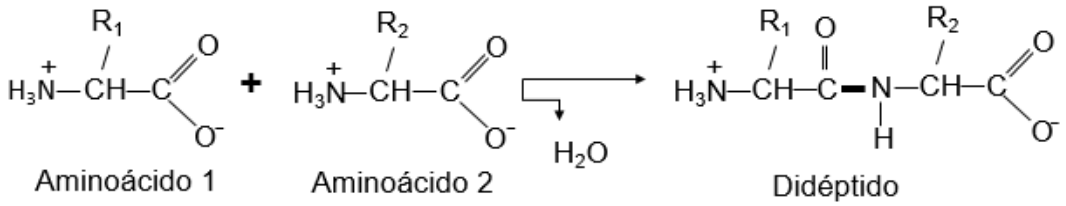

Figura 1. Concatenación de dos aminoácidos. Ri corresponde a las cadenas laterales.

La asombrosa diversidad de las miles de proteínas encontradas en la naturaleza deviene de las propiedades intrínsecas de sólo 20 aminoácidos (Ver Tabla 1) que se combinan en una cadena lineal en millones de configuraciones que aportan a cada proteína características particulares (26).

Tabla 1. Abreviaciones de los aminoácidos.

Aminoácido	Código 3 letras	Código 1 letra
Alanina	Ala	A
Cisteina	Cys	C
Áspartato	Asp	D
Glutamato	Glu	E
Fenilalanina	Phe	F
Glicina	Gly	G
Histidina	His	H
Isoleucina	Ile	I
Lisina	Lys	K
Leucina	Leu	L

Aminoácido	Código 3 letras	Código 1 letra
Metionina	Met	M
Asparragina	Asn	N
Prolina	Pro	P
Glutamina	Gln	Q
Arginina	Arg	R
Serina	Ser	S
Treonina	Thr	T
Valina	Val	V
Triptófano	Trp	W
Tirosina	Tyr	Y

1.1.1 Niveles estructurales de las proteínas

La arquitectura de las proteínas se subdivide en cuatro niveles de organización. Una serie de aminoácidos unidos por enlaces peptídicos forman una cadena polipéptidica, y cada unidad de aminoácido en un polipéptido se denomina un residuo. Actualmente, se conocen las secuencias de aminoácidos completas de más de 2 000 000 de proteínas. La secuencia de aminoácidos de una proteína es conocida como su **estructura primaria**. Tal estructura está codificada completamente en el genoma de cada organismo (27). Algunas zonas de esta cadena adquieren una disposición espacial regular y ampliamente observadas en diversas proteínas, que son denominadas **estructuras secundarias** (hélices alfa, hojas β, etc.) que se debe a las interacciones entre residuos próximos en la secuencia de aminoácidos (28). Esta estructura, a su vez puede plegarse en una disposición tridimensional conocida como **estructura terciaria** en la que las interacciones se pueden dar entre residuos que están alejados en la secuencia primaria y que se encuentran en estructuras secundarias diferentes (29). Por último existe la posibilidad de que varias cadenas proteicas o subunidades (que pueden ser iguales o diferentes) se asocien en complejos tridimensionales que componen la **estructura cuaternaria** de las proteínas.

Esta asociación de diversas unidades es el determinante final de la función de la proteína (30).

1.1.2 Dominios de proteínas

Algunas familias de proteínas son definidas por determinados fragmentos de la cadena de aminoácidos que son críticos para la función propia de la proteína (5). Estos fragmentos, a menudo se pliegan de forma autónoma respecto al resto de la proteína. En este sentido, aquellas regiones de una cadena polipeptídica que se pueden plegar de forma estable e independiente y a la vez dotar de características funcionales a las proteínas, son llamados dominios. En muchos casos, un dominio de una proteína grande conservará su estructura tridimensional incluso cuando se separa del resto de la cadena polipeptídica. Diferentes dominios a menudo tienen funciones distintas, tales como la unión de moléculas pequeñas o la interacción con otras proteínas. Las proteínas pequeñas por lo general sólo tienen un dominio en tanto es común encontrar múltiples dominios en proteínas de gran tamaño (31).

1.1.3 La base de datos de dominios de proteínas PFam

Pfam posee una colección completa de familias de dominios de proteínas. Esta base de datos contiene dos tipos de distribuciones, *Pfam-A* correspondiente a proteínas manualmente curadas y *Pfam-B* a donde pertenecen las proteínas generadas automáticamente, derivadas de clústeres producidos por el algoritmo ADDA (32). En Pfam, las familias son conjuntos de secuencias de aminoácidos que expresan un mismo domino, lo que implica que guardan relación funcional y estructural entre las mismas (11). Actualmente PFam contiene 16306 familias de dominios de proteínas manualmente curados.

1.2 Interacción entre proteínas

En una célula viviente, las proteínas realizan una serie dinámica de funciones, pero para ello actúan recíprocamente con otras proteínas y componentes celulares. Por consiguiente, la información sobre la interacción entre pares de proteínas puede conllevar al descubrimiento de posibles funciones de proteínas no conocidas (19).

La identificación de las redes de interacción de proteínas ha recibido considerable atención en los últimos tiempos. Los enfoques bioquímicos disponibles actualmente utilizados para detectar interacciones proteína-proteína conllevan mucho tiempo y un trabajo intensivo. Por ello la predicción de PPI actualmente es un tema que recibe especial atención en Bioinformática, campo interdisciplinario donde se proveen tecnologías computacionales útiles para la gestión y el análisis de datos (33).

1.2.1 Bases de datos de interacciones entre dominios de proteínas (DDI)

Los métodos computacionales que predicen interacciones biológicas solo pueden ser juzgados cuando están soportados por casos comprobados por experimentos *en vivo* o *in vitro* (34). De numerosos esfuerzos han emergido un conjunto de bases de datos donde, por diversos enfoques, se han agrupado casos de dominios para los que se ha evidenciado la ocurrencia de interacción, así como casos de dominios que se ha probado no lo hacen; comprobados por medios de experimentación científica. Algunas de ellas aparecen a continuación:

1.2.1.1 Dominios de interacción en 3D (3did)

La base de datos 3DID contiene una colección de interacciones basadas en dominios de proteínas cuya estructura 3D es conocida (11). 3did recoge y clasifica todas las plantillas estructurales de interacciones dominio-dominio que se encuentran en el Banco Mundial de Proteínas (PDB) (35), con todos los detalles moleculares de tales interacciones (6). En 3did, por cada par de dominios que interactúan, se agrupan las plantillas estructurales correspondientes en base a la interfaz de interacción, con el fin de caracterizar los diferentes modos de interacción entre el mismo par de dominios.

1.2.1.2 iPfam

La base de datos iPfam cataloga las interacciones de los dominios de proteínas de Pfam que están basadas en las estructuras 3D conocidas que se encuentran en el PDB, proporcionando datos de la interacción a nivel molecular.

Anteriormente, los datos de dominio de la interacción entre dominios de iPfam se encontraban integrados dentro de la base de datos y el sitio web de Pfam. Actualmente, estos datos se separaron a una base de datos independiente, con el fin de mejorar el acceso a dicha información a través de una separación más clara entre las familias de proteínas y los conjuntos de datos de las interacciones (7).

1.2.1.3 Negatome

Negatome es una colección de pares de dominios que tienen pocas probabilidades de estar envueltos en interacciones físicas directas. La base de datos contiene pares de dominios derivados de la curación manual a partir de la literatura y mediante el análisis de complejos de proteínas con estructura 3D conocida (8).

Negatome se subdivide en varios conjuntos de pares donde para su obtención se emplean distintos niveles de filtrado. Entre ellos *"Combined PFam"* es aquel que contiene combinaciones de dominios de proteínas sin interacción; obtenido a través de la aplicación de todos los filtros. A continuación, se muestra la Tabla 2 que contiene una breve descripción de este conjunto así como de cada conjunto de Negatome del cual se deriva el mismo:

Tabla 2. Orígenes de los datos de Negatome.

Base de datos	**Derivado de…**	**Descripción**
Combined Pfam	Manual-PFAM y PDB-PFAM	Contiene combinaciones de dominios de proteínas que no interactúan.
Manual-PFAM	Manual-stringent	Contiene pares de dominios de PFam encontrados en la base de datos *Manual* filtrados usando iPfam y 3did.
Manual-stringent	Manual	La base de datos *Manual* filtrada contra la base de datos *IntAct* (36).
Manual	"Manual literature annotation"	Contiene datos de la literatura anotados manualmente que describen la falta de interacción entre proteínas.
PDB-PFAM	PDB-stringent	Contiene pares de dominios de PFam que no interactúan encontrados en el mismo complejo estructural, filtrados

		usando iPFam y 3did.
PDB-stringent	PDB (35)	La base de datos PDB filtrada contra la base de datos IntAct.
PDB	La base de datos del PDB	Contiene pares de proteínas que son miembros de al menos un complejo estructural, pero que no interactúan directamente.

1.3 Descriptores moleculares o numéricos

Es posible identificar proteínas que presentan características "similares" basándose en el análisis de información obtenida de las secuencias (13). Una forma de representación de la información contenida en una secuencia de aminoácidos es mediante descriptores numéricos.

Un descriptor es el resultado final de un procedimiento lógico y matemático que transforma información química codificada dentro de una representación simbólica de una molécula en un número útil o el resultado de algún experimento estandarizado (18).

El programa ProtDCal (16) fue introducido como una herramienta para el cálculo de descriptores de proteínas con propósito general. ProtDCal genera atributos para secuencias de aminoácidos de tipo no dimensional o basados en composición de aminoácidos (0D) y unidimensional que toman en cuenta las vecindades lineales de los aminoácidos (1D). Los mismos son obtenidos empleando múltiples ponderaciones, de naturaleza químico – física, para cada residuo de una proteína. Los valores de los índices de cada residuo son luego modificados mediante un operador de vecindad determinado que puede considerar tanto vecindades locales (residuos adyacentes) como globales (el resto de la proteína). Luego son extraídos grupos de residuos de la proteína según sus tipos o posiciones en la cadena. El cálculo final de los descriptores se realiza mediante operadores de agregación como normas, estadígrafos de tendencia central, dispersión, así como medidas de teoría de información sobre los valores de los índices en los diferentes grupos de residuos (Ver figura 2).

La finalidad de los atributos generados es su uso en el desarrollo de relaciones cuantitativas estructura-propiedad en proteínas empleando técnicas de

aprendizaje automatizado tales como las Máquinas de Vectores de Soporte, Redes Neuronales Artificiales (ANN), k-Vecinos más Cercanos (k-NN), etc.

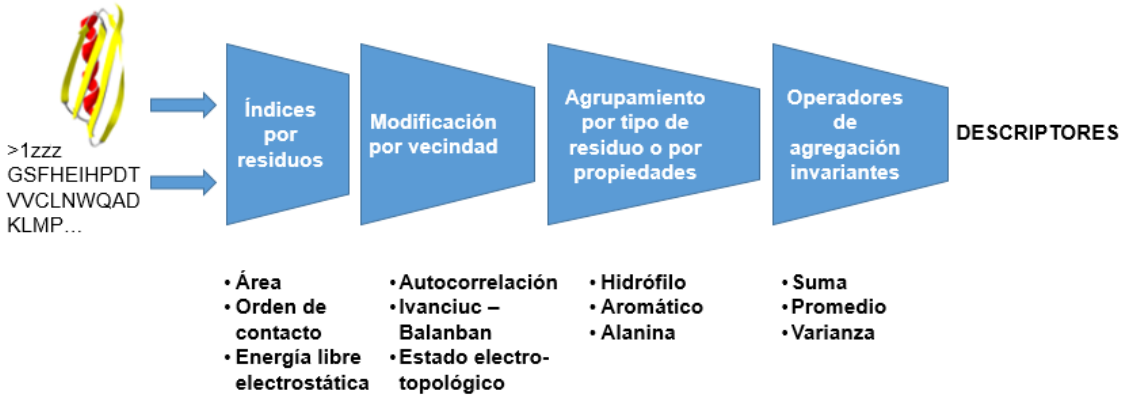

Figura 2. Flujo correspondiente al cálculo de descriptores de ProtDCal.

Estos descriptores poseen la característica de no emplear alineamientos entre secuencias de aminoácidos por lo que pueden ser aplicados independientemente de los niveles de homología que existan entre las secuencias de la base de conocimientos y las que se desean predecir.

1.4 Métodos computacionales para la predicción de PPI

A consecuencia del alto costo que sufren los métodos tradicionales de detección de PPI y del desarrollo vertiginoso de las ciencias de la computación, han ido en aumento los estudios encaminados a desarrollar herramientas computacionales capaces de identificar eficazmente tales interacciones. En la literatura es posible encontrar métodos contemporáneos que emplean descriptores numéricos de proteínas unidos a SVM en la modelación de PPI (19). Estos métodos constituyen las herramientas públicas de mayor acceso para predecir PPI a partir de secuencias de aminoácidos. En un estudio realizado por Yungki Park en 2009 (19) son comparados tres métodos basados en información obtenida de las secuencias de aminoácidos de las proteínas, donde dos de ellos utilizan SVM como técnica de Aprendizaje Automático para clasificar las interacciones:

1.4.1 PIPE

El motor de predicción de interacciones proteína-proteína, denominado PIPE por sus siglas en inglés, es una herramienta capaz de predecir interacciones

proteína-proteína para cualquier par objetivo de las proteínas de la levadura *Saccharomyces cerevisiae* a partir de su estructura primaria y sin la necesidad de cualquier información adicional o predicciones acerca de las proteínas (9).
Para un par de proteínas, PIPE busca las coocurrencias de sus subsecuencias en pares de proteínas que se conoce que interactúan (19). PIPE mostró una sensibilidad del 61% para la detección de cualquier interacción de proteínas de levadura con 89% de especificidad y una precisión global de 75% (20).

1.4.2 SPPS

La herramienta ''Búsqueda de ligandos de proteínas basados en la secuencia'' (SPPS por sus siglas en inglés) ha sido desarrollada para explorar ligandos de proteínas que interactúan, mediante la búsqueda a través de un gran repertorio de proteínas de muchas especies. SPPS proporciona una base de datos que contiene más de 60.000 secuencias de proteínas con anotaciones y un motor de búsqueda de proteína-ligando en dos modos: de consulta, que busca todos aquellos ligandos que muestran alta probabilidad de interacción con una proteína dada y de consulta múltiple, donde para un par de proteínas se determina la probabilidad de interacción (21).
El método fue desarrollado usando únicamente información obtenida de las secuencias simples de las proteínas. Se desarrolló basándose en el algoritmo de Aprendizaje Automático SVM, combinado con una nueva función kernel y un conjunto de elementos establecidos para describir los aminoácidos. SPPS hace uso de la probabilidad arrojada por el método SVM para mostrar posibles interacciones entre pares de proteínas que se encuentran en una serie de bases de datos de proteínas que abarcan un grupo de especies. Cinco modelos que incluyen *‘‘Homo sapiens’’*, *‘‘Mus musculus’’*, *‘‘Caenorhabditis elegans’’*, *‘‘Drosoph- ila melanogaster’’* y *‘‘Saccharomyces cerevisiae’’* fueron construidos sobre la base de la recolección de aquellas PPIs que mostraron una buena exactitud.

1.4.3 Pred-PPI

Este método fue creado basándose en las secuencias simples de aminoácidos de las proteínas y combinando una forma de representación donde se emplean

auto-covarianza (AC) y SVM. La AC responde a la interacción entre residuos que se encuentran a una cierta distancia dentro de una misma secuencia; por lo que este procedimiento adecuadamente toma en cuenta el efecto de la vecindad. Para su desarrollo, los residuos de aminoácidos fueron traducidos a valores numéricos de forma tal que representasen sus propiedades físico-químicas. Posteriormente estas secuencias numéricas fueron analizadas por AC basándose en el cálculo de covarianza. Por último, un modelo SVM fue construido usando como entrada los vectores de las variables de AC (22).

1.5 Aplicación de SVM en la predicción de PPI

En la literatura, los métodos computacionales de detección de PPI que involucran reconocimiento de patrones a partir de las secuencias de aminoácidos, son en su mayoría basados en la técnica de aprendizaje automatizado SVM. En la tesis doctoral de Sylvain Pitre (9): *"PIPE: Un motor de predicción de interacciones proteína-proteína basado en la reapariciones de pequeñas secuencias polipetídicas entre pares de proteínas con interacción reportada",* el autor presenta un nuevo método basado en las secuencias de aminoácidos de las proteínas para identificar PPI. PIPE no emplea técnicas de aprendizaje automatizado para predecir interacciones entre pares de proteínas; sin embargo, en esta tesis se menciona que uno de los enfoques para detectar PPI es basado en la habilidad de una SVM para automáticamente reconocer los patrones de correlación de las secuencias y las subestructuras en pares de proteínas con interacción. A lo largo del trabajo se citan varios modelos de predicción de PPI (37) (39) (41), todos ellos basados en SVM, no evidenciándose el uso de otro algoritmo de predicción.

La afirmación de Sylvain Pitre está basada, entre otros motivos, en la existencia de estudios donde se evidencia el uso de SVM para clasificar propiedades y funciones de proteínas, algunos de ellos son:

- ✓ Predecir interacciones proteína-proteína a partir de la estructura primaria (43).
- ✓ SVM-Prot: una herramienta web basada en Máquina de Vectores de Soporte para la clasificación funcional de una proteína a partir de su estructura primaria (45).

- ✓ SVMCRYS: un enfoque de SVM para la predicción de la propensión de las cristalización de proteínas a partir de la secuencia de las proteínas (47).

1.6 Conclusiones parciales

En este capítulo se realiza un resumen de los conceptos relacionados con la interacción entre proteínas, enmarcando el trabajo a la predicción de interacción entre pares de dominios de proteínas. Se identifican 3did, iPFam y Negatome como bases de datos que contienen pares de dominios de proteínas con interacción y sin interacción reportada, útiles para el estudio a realizar. Se describen los descriptores numéricos como un instrumento para representar la información contenida en la secuencia de aminoácidos de una proteína en un número de utilidad. Se identifican los métodos computacionales más recientes para predecir PPI a partir de secuencias de aminoácidos: SPPS, Pred-PPI y PIPE.

Capítulo 2: Métodos a emplear en la solución del problema

En el presente capítulo se resumen los métodos empleados en el desarrollo del clasificador de DDI. Se describe el funcionamiento de la técnica de clasificación supervisada *Máquinas de Vectores de Soporte*. Se muestran los métodos de selección de atributos agrupados por categoría según el tipo de selección que realizan. Se describen algunas técnicas de agrupamiento útiles para separar los casos en conjuntos de entrenamiento y prueba. Por último, se indican las métricas de evaluación de desempeño a emplear en la modelación.

2.1 Clasificación supervisada

La clasificación supervisada es la rama del Reconocimiento de Patrones que descubre las relaciones entre las características de los objetos y un conjunto predefinido de clases, con la finalidad de predecir la clase a la que pertenecen objetos no vistos (38). Debido a la gran diversidad de problemas en los que se aplica el reconocimiento de patrones, existe una larga colección de técnicas para encontrar estas relaciones, y una gran cantidad de algoritmos de predicción de clases basados en ellas (40). El entrenamiento de los métodos y la predicción final seguirán un esquema de combinación propio del método empleado.

2.1.1 Máquinas de Vectores de Soporte

Las Máquinas de Vectores de Soporte, SVM por sus siglas en inglés, son una técnica muy útil para resolver problemas de clasificación (42). Desde la década de 1990, han sido gradualmente aplicadas a problemas de las ciencias naturales y sociales, incrementándose aún más su uso en el siglo 21 (44).
Formalmente SVM es un algoritmo de aprendizaje supervisado que aprende un conjunto de instancias distribuidas de forma idéntica e independiente $\{(x_1, y_1),..., (x_N, y_N)\}$, donde $y \in \{-1, 1\}$ corresponde al valor de las clases binarias a las cuales responden los datos. (46)

2.1.1.1 Clasificación lineal

Un clasificador binario de entrada lineal tiene la forma:

$$\omega^T - b = 0$$

Ecuación 1. Clasificador binario de entrada lineal.

Donde ω y b son los parámetros del modelo que corresponden al vector normal y a la intercepción del límite de decisión, respectivamente (Ver figura 3).

Se asume que los casos son linealmente separables, si existe un hiperplano que separe completamente los casos pertenecientes a dos clases. En este caso, se buscan dos hiperplanos de manera tal que no existan puntos entre ellos y se maximiza su distancia. El área entre los hiperplanos se conoce como el margen.

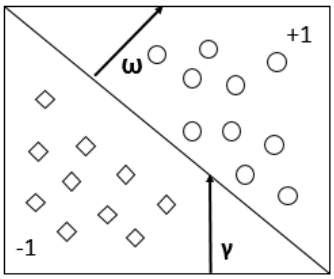

Figura 3. Himperplano de una SVM.

Por consiguiente, de forma simple y lineal, una MVS es un hiperplano que separa un conjunto de casos positivos de otro conjunto de casos negativos con el máximo margen. La distancia entre los dos planos es $\left\|\frac{2}{\omega}\right\|$; por lo tanto, al minimizar ω se converge al máximo margen, que a su vez converge a una adecuada generalización. La fórmula para la salida de una MVS lineal es:

$$\widehat{\gamma}_i = (\omega^T x_i + b)$$

Ecuación 2. Fórmula para la salida de una MVS lineal.

Donde x_i es el $i - ésimo$ caso de entrenamiento. Teniendo en cuenta lo anterior, las condiciones para que los casos no entren en el margen son las siguientes:

$$(\omega^T x_i - b)\,\gamma_i \geq 1\; para\; y_i = 1,$$
$$(\omega^T x_i - b)\,\gamma_i \leq -1\; para\; y_i = -1$$

Ecuación 3. Condiciones para la no entrada en el margen.

Estas condiciones pueden abreviarse en:

$$\gamma_i = (\omega^T x_i - b) \geq 1, i = 1, \ldots, N$$

Ecuación 4. Simplificación de la Ecuación 3.

La optimización se encuentra sujeta a estas restricciones, y se busca el hiperplano de decisión óptimo con:

$$\underset{\omega, b}{argmin} \frac{1}{2}\|\omega\|^2$$

Ecuación 5. Hiperplano de decisión óptimo.

El margen además es igual a la distancia del hiperplano al más cercano de los casos positivos y negativos (Ver figura 4).

Para obtener la formulación doble del problema se introducen primero los multiplicadores de Lagrange α_i para incluir las restricciones en la función objetivo:

$$\underset{\omega, b}{argmin} \quad \underset{\alpha_i > 0}{max} \left(\frac{1}{2}\|\omega\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(\omega^T x_i - b) - 1] \right)$$

Ecuación 6. Función objetivo.

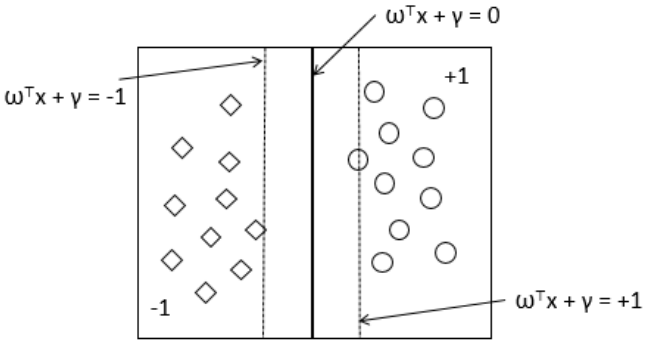

Figura 4. Límite de decisión de SMV de margen duro.

Los α_i correspondientes a los vectores que no son de soporte se igualan a 0, ya que no hacen la diferencia en encontrar el punto límite de la función objetivo expandida. Se denota la función objetivo en la Ecuación 6 como $L(\omega, b, \alpha)$. Igualando las derivadas de ω y b a cero, se obtiene:

$$\frac{\vartheta L}{\vartheta \omega_i} = \omega_i - \alpha_i y_i x_i = 0$$

$$\frac{\vartheta L}{\vartheta b} = \sum_{i=1}^{N} \alpha_i y_i = 0$$

Ecuación 7. Derivadas de ω y b.

Hasta aquí, la solución es una combinación lineal de los vectores de entrenamiento:

$$\omega = \sum_{i=1}^{N} \alpha_i y_i \, x_i$$

Ecuación 8. Combinación lineal de los vectores de entrenamiento.

Insertando la misma dentro de la función objetivo en la Ecuación 6, se puede expresar el problema de optimización solamente en términos de α_i. Se definen el problema con los multiplicadores α_i:

$$\max_{\alpha_i} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \, x_i x_j$$

Ecuación 9. Función objetivo con los multiplicadores.

Sujeto a (para cualquier i=1…N) $\alpha_i \geq 0$,

Para la minimización de b, la restricción adicional es:

$$\sum_{i=1}^{N} \alpha_i \, y_i = 0$$

Ecuación 10. Minimización de b.

Eventualmente, solamente algunas α_i no serán iguales a 0. Los correspondientes x_i son los vectores de soporte que se ubican en el margen. Entonces, para los vectores de soporte, $\gamma_i(\omega x_i - b)$ es igual a 1. Reordenando la ecuación, se obtiene el término bias:

$$b = (\omega^T x_i - \gamma_i)$$

Ecuación 11. Término bias b.

2.1.1.2 Clasificación no lineal

No todos los problemas de clasificación son linealmente separables. Algunos casos de una clase pueden estar mezclados con elementos de la otra clase. Un clasificador lineal no es capaz de eficientemente aprender un modelo a partir de estos datos. Aun si las clases son separables, la dependencia al margen hace a una máquina de vectores de soporte lineal sensible a unos pocos puntos ubicados cerca del límite (46) (Ver figura 5).

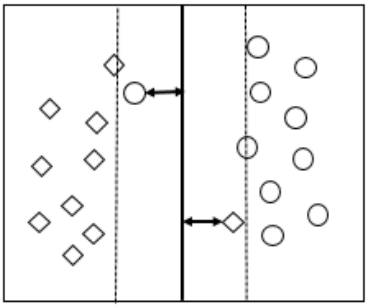

Figura 5. Máquinas de vectores de soporte no lineal.

Para lidiar con estos casos, se utilizan las variables de holguras, las cuales miden el grado en que un caso se desvía del margen. La optimización se convierte entonces en un compromiso entre maximizar el margen y controlar las variables de holgura. Un parámetro C balancea este compromiso, el cual es denominado parámetro de costo.

La clasificación no lineal con variables de holgura se convierte en:

$$(\omega^T x_i + b) \geq 1 - \xi_i \; si \; y_i = 1,$$
$$(\omega^T x_i + b) \leq -1 + \xi_i \; si \; y_i = -1$$

Ecuación 12. Clasificación no lineal con variables de holgura.

La primera forma de optimización es minimizar $\|\omega\|$ y la cantidad de desviación descrita por las variables de holgura, sujetas a las restricciones:

$$\min \frac{1}{2}\|\omega\|^2 + C\sum_i \xi_i$$

Ecuación 13. Restricciones.

La correspondiente Lagrange, análoga a la Ecuación 6 es:

$$\underset{\omega,\xi,b}{argmin} \quad \underset{\alpha,\beta}{max}\left(\frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i[y_i(\omega^T x_i - b) - 1 + \xi_i] - \sum_{i=1}^{N}\beta_i \xi_i\right)$$

Ecuación 14. Función objetivo análoga a la Ecuación 6.

Con $\alpha_{i,}\beta_i \geq 0$. Tomando las derivaciones parciales se derivan relaciones similares a la Ecuación 7, con la igualdad adicional $C - \beta_i - \alpha_i = 0$, que junto con $\beta_i \geq 0$ implican que $\alpha_i \leq C$.

Así, la formulación doble es similar a la de la clasificación lineal:

$$\max_{\alpha_i} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \, x_i x_j$$

Ecuación 15. Formulación doble.

Sujeto a $0 \leq \alpha_i \leq C, i, \dots, N$,

$$\sum_{i=1}^{N} \alpha_i \, y_i = 0$$

Ecuación 16. Restricciones de la Ecuación 15.

Las variables de holgura distintas de cero se producen cuando αi alcanza su valor más alto. El parámetro de costo C actúa como regulador: si el costo de los errores de clasificación es alto, un modelo más preciso es buscado con mayor complejidad, es decir, con un mayor número de vectores de soporte.

2.1.1.3 Funciones Kernel

Lo que hace a las Máquinas de Vectores de Soporte más ponderosas es que no están restringidas a espacios de decisión lineal. La doble formulación permite una proyección no lineal donde se mapean las instancias desde un espacio de entrada hacia un espacio dimensional embebido más grande, que es denominado espacio de rasgos en este contexto (48). La idea principal es remplazar el producto $x_i x_j$ en la función objetivo en la Ecuación 15 por una función que retenga muchas de las propiedades del producto, pero sin tener que ser lineal. Esta función es llamada kernel.

Se embeben los casos con una función ∅ que clasifica los puntos como:

$$y_i \, (\, \omega^T \emptyset(x_i) + b) \, \geq 1 - \xi_i, \xi_i \geq 1, i = 1, \dots, N$$

Ecuación 17. Función kernel.

Con estas restricciones se busca la solución del siguiente problema de optimización:

$$min \frac{1}{2} \|\omega\|^2 + C \sum_{i=1}^{n} \xi_i$$

Ecuación 18. Problema de optimización.

El objetivo es idéntico al de margen suave Ecuación 13.

Igual que anteriormente, se introducen multiplicadores Lagrange para acomodar las restricciones del problema. Las derivaciones parciales en ω, b y ξ definen un punto extremo de Lagrange, con la cual la formulación doble se convierte en el problema de programación cuadrático siguiente:

$$\max_{\alpha_i} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \, K(x_{i,} x_j)$$

Ecuación 19. Formulación doble con funciones kernel.

Sujeto a

$0 \leq \alpha_i \leq C, i, \ldots, N,$

$$\sum_{i=1}^{N} \alpha_i \, y_i = 0$$

Ecuación 20. Restricciones de la Ecuación 19.

La función $K(x_{i,} x_j) = \emptyset(x_i)^T \emptyset(x_j)$ es la función kernel, el producto escalar del espacio embebido.

En la formulación doble la función kernel realiza el cálculo de lo embebido por sí misma. Se utiliza la función kernel directamente para encontrar el óptimo y, posteriormente clasificar nuevos casos. El espacio donde se embeben los casos puede ser de dimensiones infinitas.

Algunos kernels importantes son listados en la Tabla 3.

El kernel polinomial tiene un parámetros bias c y un parámetro grado d.

El kernel *Radial Basis Function* opera en un espacio de infinitas dimensiones. El parámetro Gamma (y) controla el radio de la función base.

Tabla 3. Kernels comunes.

	Lineal	Polinomial	Radial Basis Function
$K(x_{i,} x_j)$	${x_i}^T x_j$	$({x_i}^T x_j + c)^d$	$\exp(-\gamma \left\|(x_i - x_j\right\|^2)$

2.1.1.4 Metodología para el uso de SVM

Para llevar a cabo el entrenamiento de un modelo haciendo uso de SVM se sigue una metodología sencilla que incluye los siguientes pasos:

1- Normalizar los atributos.

2- Seleccionar los parámetros más apropiados.
   - Parámetro *C* (coste, complejidad): Un valor grande de coste trae un consigo una mayor complejidad y puede conllevar a un posible sobreaprendizaje. Mientras que un valor pequeño conlleva a una complejidad menor y converge a una mejor generalización.
   - Definir el Kernel a emplear:
     - Polinómico: exponente (d)
     - R: gamma
3- Construir el modelo con los mejores parámetros y todo el conjunto de entrenamiento.

2.1.1.5 Metodología para el ajuste de los parámetros

Un paso determinante en el uso de SVM es encontrar para qué valores de los parámetros la exactitud del clasificador aumenta. Por tal motivo es necesario realizar una exploración de parámetros según el kernel a utilizar, para ello también se sigue una metodología.

Para el Kernel Gaussiano (RBF kernel) los parámetros a ajustar son C y gamma. La técnica consiste en realizar una búsqueda en grid (enrejado), haciendo validación cruzada para cada combinación de valores (C, gamma).

Se define un rango de búsqueda (logarítmico):

- $C = 2^{-5}, 2^{-3}, 2^{-1}, 2^{3}, \ldots, 2^{15}$, (21 valores en total)
- Gamma = $2^{-15}, 2^{-13}, \ldots, 2^{3}$, (19 valores en total)

Una vez identificada una buena zona, se hace una búsqueda en grid más fina en la zona deseada con pasos de 0.25 en el valor del exponente. Por último se pasa a aprender el modelo con los parámetros finales y todo el conjunto de entrenamiento.

2.2 Métodos para la selección de atributos

Se ha mostrado empíricamente en diferentes estudios que rasgos redundantes o no discriminatorios reducen la habilidad de los clasificadores de aprender los límites entre los datos de diferentes clases (49). Por esta razón, la identificación de los rasgos o atributos más relevantes es extremadamente importante en tareas de clasificación, para aumentar la exactitud del

clasificador y al mismo tiempo reducir el costo computacional; así como para entender el significado relativo de los rasgos.
Los parámetros óptimos de un modelo generado a partir del conjunto de todos los atributos no siempre son los mismos que los de un modelo generado usando los rasgos seleccionados al hacer selección de atributos (50).

Los esquemas de selección de atributos son a menudo divididos en dos categorías: (i) *Métodos de filtrado*, los cuales usan solo las características de los datos de entrenamiento sin hacer referencia al algoritmo de aprendizaje que será finalmente usado, (ii) *Métodos Wrapper*, que hacen uso del comportamiento de un clasificador determinado para seleccionar los rasgos (51).

2.2.1 Técnicas de filtrado

Estas técnicas les dan importancia a los rasgos buscando solo en las propiedades básicas de los datos. En la mayoría de los casos un valor que representa la relevancia de un atributo es calculado, y los rasgos que tienen valores más bajos son eliminados. Posteriormente la data resultante es utilizada como entrada al algoritmo de clasificación. Estas técnicas son independientes del algoritmo de clasificación y tienen como ventaja que son fáciles de aplicar a datas de gran dimensión por su sencillez y rapidez computacional. Como desventajas tiene que ignoran las dependencias entre los rasgos e ignoran la interacción con el clasificador (51).

Ganancia de información

La Ganancia de Información (GI) es una métrica derivada de la Teoría de Información, que determina el contenido de información aportado por una variable Y para describir la información de una variable X (52).
Esta métrica se formula como la diferencia de la entropía de información de la variable X y la entropía condicional de X dado una segunda variable Y (53).

$$IG_c(X|Y) = H(X) - H_c(X|Y)$$

Ecuación 21. Ganancia de Información.

Siendo X la variable de clases del problema (ej. dominios con interacción y dominios sin interacción) el primero de los términos representa la información

total necesaria para describir la distribución de clases del conjunto de casos empleado. En tanto el término condicional representa la información faltante para describir la variable de clase conociendo el descriptor (variable independiente) Y (53). Las formulaciones para cada uno de estos términos son las siguientes:

$$H(X) = -\sum_{i} P(x_i) log_2(P(x_i)) \;\; i = 1, 2$$

Ecuación 22. Entropía de información de la variable X.

$$H_c(X|Y) = -\sum_{j} P_c(y_j) \sum_{i} P_c(x_i|y_j) log_2(P_c(x_i|y_j))$$

Ecuación 23. Entropía condicional de X dado una segunda variable Y.

Donde $P(x)$ es la probabilidad a priori de cada una de las clases calculada como la fracción del número de casos de la clase X en el número total de casos del conjunto de datos. $Pc(x|y)$ es la probabilidad condicional de la clase X dado determinados valores del descriptor Y, se obtiene como la fracción de casos de la clase X en un conjunto de casos seleccionado de acuerdo a sus valores del descriptor Y. Finalmente $Pc(y)$ representa la probabilidad de un conjunto de casos seleccionados según sus valores del descriptor Y, se obtiene como el cociente entre el número de casos en el subconjunto y el número de casos en el conjunto total de datos. Esta última probabilidad permite obtener un promedio ponderado de la entropía condicional de los diferentes subconjuntos definidos por los valores del descriptor Y, resultando en la entropía condicional de la variable de clase dado un descriptor.

La entropía de la variable de clase y a su vez la Ganancia de Información dado un descriptor, son magnitudes propias del conjunto de casos en que se evalúen, por lo que no es posible establecer un criterio uniforme para seleccionar atributos en diversos conjuntos de datos, basándose directamente en la GI.

En el presente trabajo se emplea la Ganancia de Información normalizada (NIG) con el contenido de información (entropía) de la variable clase:

$$NIC_c(X|Y) = \frac{IG_c(X|Y)}{H(X)}$$

Ecuación 24. Ganancia de Información normalizada.

De esta forma se seleccionan descriptores, cuyos valores de GI superen un determinado por ciento del contenido de información total de la variable de clase.

Técnica de clustering "*Single-linkage"*

En el *clustering* "*single-link" (*enlace único o agrupación de un solo vínculo), la similitud de dos clústeres se define por la similitud de sus miembros más similares (54). Este criterio de combinación de un solo enlace es local y en él se le presta atención únicamente a la zona en la que los dos grupos se acercan más el uno al otro. Las otras partes más distantes del clúster y la estructura general de los clústeres no son tenidas en cuenta para hacer la agrupación. El resultado de la formación de conglomerados con este algoritmo es que miembros de conglomerados distintos no pueden encontrarse por debajo de un determinado valor de corte según la medida de diversidad empleada (lo contrario se aplica si se emplea una medida de similitud).

Coeficiente de correlación simple por rangos de Spearman

Un coeficiente de correlación, mide el grado de relación o asociación existente generalmente entre dos variables aleatorias (55).

El Coeficiente de correlación de Spearman permite medir la correlación de dos variables sin asumir a priori una relación lineal entre las mismas, utilizando para ello la clasificación por rangos. El trabajo por rangos limita además el efecto de mediciones extremas en el conjunto de datos, las cuales afectan al coeficiente de correlación de Pearson común. Se rige por las reglas de la correlación simple de Pearson, y las mediciones de este índice oscilan entre -1 y +1, indicándonos asociaciones negativas o positivas respectivamente, en donde 0 significa no correlación pero no independencia.

La ecuación utilizada en este procedimiento es la siguiente:

$$r_s = 1 - \frac{6\sum d^2}{N^3 - N}$$

Ecuación 25. Coeficiente de correlación de Spearman.

donde:

r_s = es el coeficiente de correlación de Spearman.

d^2 = es la diferencia existente entre los rangos de las dos variables, elevadas al cuadrado.

N = es el tamaño de la muestra expresada en parejas de rangos de las variables.

Pasos a seguir:

1. Clasificar en rangos cada medición de las observaciones.
2. Obtener las diferencias de las parejas de rangos de las variables estudiadas y elevadas al cuadrado.
3. Efectuar la sumatoria de todas las diferencias al cuadrado.
4. Aplicar la ecuación.
5. Calcular los grados de libertad (gl), donde gl = número de parejas - 1. Solo se utilizará cuando la muestra sea mayor a 10.
6. Comparar el valor r calculado con respecto a los valores críticos de la tabla de valores críticos de t de Kendall en función de probabilidad.
7. Decidir si se acepta o rechaza la hipótesis.

2.2.2 Técnicas Wrapper

A diferencia de las técnicas de filtrado, en los métodos Wrapper, un algoritmo de aprendizaje se enfrenta al problema de seleccionar un subconjunto de características relevantes sobre los que centrar la atención, mientras el resto son ignoradas. En esta configuración, el resultado del procedimiento de búsqueda corresponde al espacio de los posibles subconjuntos de rasgos, y varios subconjuntos de rasgos son generados. Para lograr el mejor desempeño posible con un algoritmo de aprendizaje en particular en un conjunto de entrenamiento determinado, este método de selección de atributos considera cómo el algoritmo y el conjunto de entrenamiento interactúan (56).

En el popular método originalmente se propuso, para los dominios de expresión de genes, utilizar los pesos de las variables de la formulación SVM para

descartar rasgos con pesos pequeños; lo cual ha sido aplicado con éxito en diversos estudios (57).

Con el fin de elegir el mejor subconjunto de atributos, se necesita realizar una búsqueda que tomará tiempo exponencial, razón por la cual se hace uso de la heurística. En los enfoques Wrapper diferentes tipos de heurísticas aleatorias son empleadas como motores de búsqueda en muchos estudios: BF, GA, PSO, etc.

Estrategias wrapper frecuentemente utilizadas son la *Selección Secuencial hacia Adelante* (Sequential forward selection o SFS) y la *Eliminación Secuencial hacia Atrás* (Sequential backward elimination o SBE) (58). Para el primer caso, el modelo parte sin considerar variables, para luego probar cada una de ellas e incluir la más relevante en cada iteración. De la misma manera, SBE parte con todas las variables candidatas a formar parte del modelo, eliminando de forma iterativa aquellas variables irrelevantes para la clasificación.

Una desventaja de estas técnicas es que ellas tienen un alto riesgo de sobreaprendizaje más que con las técnicas de filtrado y son muy intensivas computacionalmente, especialmente si construir el clasificador tiene un costo computacional alto.

2.3 Agrupamiento de datos en conjuntos para la modelación

Es totalmente vital medir el comportamiento de un clasificador en un conjunto independiente de prueba. Cada algoritmo de entrenamiento busca patrones en los datos de entrenamiento, ejemplo, la correlación entre los atributos y la clase. Algunos de los patrones descubiertos pueden ser solo aparentes, por ejemplo, son válidos en el conjunto de entrenamiento debido a la aleatoriedad en la selección de los datos de entrenamiento de la población, pero no son válidos, o no son tan fuertes, en la población completa.

Un clasificador que confíe en estos patrones tendrá una exactitud más alta en el conjunto de entrenamiento que la que tendrá con toda la población. Solo la exactitud medida en un conjunto independiente de prueba a una estimación justa de la exactitud de la población completa. El fenómeno de confiar en

patrones que son fuertes solo en los datos de entrenamiento es llamado sobreaprendizaje.

Es natural correr un algoritmo de entrenamiento múltiples veces, y medir la exactitud de cada clasificador obtenido con diferentes configuraciones de parámetros. Por consiguiente, es natural elegir la configuración que brinde el mejor comportamiento en el conjunto de validación.

Un conjunto de casos etiquetados usados de esta forma para elegir la mejor configuración para un algoritmo es llamado conjunto de validación. Si se usa un conjunto de validación, es importante tener un conjunto de prueba final que sea independiente de ambos, conjuntos de entrenamiento y validación (59). Las configuraciones que dan el mejor desempeño en el conjunto de validación probablemente estén sobreaprendiendo el conjunto de validación. Este efecto puede ser mayor si muchas configuraciones diferentes son evaluadas. Solo un conjunto de prueba independiente puede dar una estimación justa del comportamiento de las configuraciones de parámetros seleccionadas.

En la división de los casos, en conjuntos de entrenamiento, prueba y validación existen un conjunto de algoritmos de agrupamiento que pueden ser empleados. Un algoritmo de agrupamiento intenta encontrar grupos naturales de datos basándose principalmente en la similitud y relaciones de los objetos, de forma tal que se obtenga una distribución interna del conjunto de datos mediante la partición del mismo en subconjuntos de datos. Esta partición debe lograr la homogeneidad dentro de los grupos, por ejemplo, los objetos que pertenecen al mismo grupo deben ser tan similares como sea posible, y la heterogeneidad entre grupos, por ejemplo, los objetos que pertenecen a grupos distintos deben ser tan diferentes como sea posible (60) (61).

Algoritmos de agrupamiento:

Algunos de los algoritmos para agrupar los casos en conjuntos de entrenamiento y prueba son:

Cobweb: realiza los agrupamientos caso a caso siguiendo una estrategia jerárquica. Para ello forma un árbol donde las hojas representan los objetos o grupos de objetos y el nodo raíz engloba por completo el conjunto de datos de entrada. Al inicio el árbol consiste en un único nodo raíz. Los casos se van

añadiendo uno a uno y el árbol se va actualizando en cada paso. La actualización consiste en encontrar el mejor sitio donde incluir el nuevo caso, lo cual puede requerir la reestructuración de todo el árbol o simplemente la inclusión del caso en un nodo existente. Para saber cómo y dónde actualizar el árbol se emplea una medida denominada utilidad de categoría, que mide la calidad general de una partición de casos en un segmento. Al algoritmo no hay que proporcionarle el número exacto de clústeres que se desean ya que el encuentra el número óptimo (62).

EM: Expectación-maximización (EM por sus siglas en inglés) asigna a cada caso una distribución de probabilidad de pertenencia a cada cluster. Este realiza un refinamiento muy costoso al manejar datos de alta dimensión. EM puede decidir cuántos clústeres crear basado en validación cruzada o se le pueden especificar al inicio cuantos generar. Utiliza el modelo Gaussiano finito de mezclas, asumiendo que todos los atributos son variables aleatorias independientes (63).

K-Means: (64) es uno de los algoritmos más sencillos de aprendizaje no supervisado, así como uno de los algoritmos para crear particiones más utilizado. Requiere que el número de grupos a obtener sea especificado a priori, por lo que es necesario cierto conocimiento del dominio, ya que es sensible a cómo se hizo inicialmente la partición. A partir de él se han derivado varios como el x-medias (x-means) para una estimación eficiente del número de grupos, el conjunto k-medias (batch k-means) y el k-medias incremental (incremental k-means), y su variante mejorada medias (means) (65).

FarthestFirst: El algoritmo Primero el más lejano mejora el k-medias (66). Parte de una selección aleatoria de los centros de grupos, calcula la distancia de cada instancia al centroide más cercano, y la instancia que quede más lejana del centroide más cercano es seleccionada como el centroide de un grupo. Este proceso es repetido hasta que el número de grupos sea mayor que un umbral especificado (67). Este algoritmo parte seleccionando aleatoriamente una instancia que será el centro del grupo. Se calcula la distancia entre cada instancia restante y su centro más cercano. La instancia que más alejada está del centro es seleccionada como un centro de grupo. Este proceso es repetido hasta que el número de grupos sea mayor que un umbral especificado.

2.4 Evaluación de los modelos

En la construcción de un modelo de clasificación es necesario tener en cuenta, tanto en la etapa del entrenamiento como en la etapa de prueba, los métodos y técnicas a emplear para evaluar los clasificadores que se generan.

2.4.1 Métricas de evaluación de desempeño

Cuando se evalúa un clasificador, existen diferentes formas de medir su desempeño. Para el aprendizaje supervisado, con dos posibles clases, todas las medidas están basadas en cuatro números obtenidos de la aplicación del clasificador al conjunto de prueba: TP, TN, FP y FN. Estos números son llamados verdaderos positivos (TP), falsos positivos FP, verdaderos negativos TN, y falsos negativos FN. En el marco del problema a modelar pueden corresponder a: los pares de dominios con interacción y predichos como tal (*TP*), los pares de dominios sin interacción y predichos como tal (*TN*), los pares de dominios sin interacción y predichos incorrectamente (*FP*), los pares de dominios con interacción y predichos incorrectamente (*FN*).
Estos números son la entrada a una tabla denominada Matriz de confusión 2x2 como sigue:

Tabla 4. Matriz de confusión.

		predicho	
		Positiva	Negativa
verdadero	Positiva	TP	FN
	Negativa	FP	TN

La suma de los valores de TP, TN, FP y FN es igual al número de casos. Dependiendo de la aplicación, diferentes estadísticas pueden ser calculadas a partir de estos datos, tales como:
Exactitud (%) = 100 * (TP + TN) / (TP + TN + FP + FN)
Precisión (%) = 100 * TP / (TP + FP)
Sensibilidad (%) = 100 * TP / (TP + FN)
Especificidad (%)= 100 * 1 – FP / (FP + TN)

2.4.2 Validación cruzada

Existe el consenso general de que alguna forma de validación es necesaria, ya que por ejemplo, los buenos resultados podrían estar basados meramente en la oportunidad, de la bondad de los datos de entrenamiento, o la simplicidad de los datos de prueba. Por tanto, un número de ensayos con diferentes datos de entrenamiento y prueba son necesarios, y sus resultados pueden ser promediados (49). A este procedimiento se denomina validación cruzada.

Una validación basada en un único conjunto de prueba puede lograrse al dividir el conjunto total de datos en dos conjuntos complementarios, realizar el análisis de un subconjunto (datos de entrenamiento o training set), y validar el análisis en el otro subconjunto (datos de prueba o test set), de forma que la función de aproximación sólo se ajusta con el conjunto de datos de entrenamiento y a partir de aquí calcula los valores de salida para el conjunto de datos de prueba (valores que no ha analizado antes) (68). La ventaja de este método es que es muy rápido a la hora de computar. Sin embargo, este método no es demasiado preciso debido a la variación de resultados obtenidos para diferentes datos de entrenamiento. La evaluación puede depender en gran medida de cómo es la división entre datos de entrenamiento y de prueba, y por lo tanto puede ser significativamente diferente en función de cómo se realice esta división. Debido a estas carencias aparece el concepto de validación cruzada.

Validación cruzada de K iteraciones

En la validación cruzada de K iteraciones o K-fold cross-validation los datos de muestra se dividen en K subconjuntos. Uno de los subconjuntos se utiliza como datos de prueba y el resto (K-1) como datos de entrenamiento. El proceso de validación cruzada es repetido durante k iteraciones, con cada uno de los posibles subconjuntos de datos de prueba (69). Este método es muy preciso puesto que evaluamos a partir de K combinaciones de datos de entrenamiento y de prueba, pero aun así tiene una desventaja, es lento desde el punto de vista computacional. En la práctica, la elección del número de iteraciones depende de la medida del conjunto de datos. Lo más común es utilizar la validación cruzada de 10 iteraciones (10-fold cross-validation) (53).

Validación cruzada aleatoria (Boostraping)

Este método consiste al dividir aleatoriamente el conjunto de datos de entrenamiento y el conjunto de datos de prueba. Para cada división la función de aproximación se ajusta a partir de los datos de entrenamiento y calcula los valores de salida para el conjunto de datos de prueba. El resultado final se corresponde a la media aritmética de los valores obtenidos para las diferentes divisiones (68). La ventaja de este método es que la división de datos entrenamiento-prueba no depende del número de iteraciones. Pero, en cambio, con este método hay algunas muestras que quedan sin evaluar y otras que se evalúan más de una vez, es decir, los subconjuntos de prueba y entrenamiento se pueden solapar.

Validación cruzada dejando uno fuera

La validación cruzada dejando uno fuera o Leave-one-out cross-validation (LOOCV) implica separar los datos de forma que para cada iteración tengamos una sola muestra para los datos de prueba y todo el resto conformando los datos de entrenamiento. La evaluación viene dada por el error, y en este tipo de validación cruzada el error es muy bajo, pero en cambio, a nivel computacional es muy costoso, puesto que se tienen que realizar un elevado número de iteraciones, tantas como N muestras se tengan y para cada una analizar los datos tanto de entrenamiento como de prueba (70).

Cálculo del error

La evaluación de las diferentes validaciones cruzadas normalmente viene dada por el error obtenido en cada iteración, ahora bien, por cada uno de los métodos puede variar el número de iteraciones, según la elección del diseñador en función del número de datos total.

Error de la validación cruzada de K iteraciones

En cada una de las k iteraciones de este tipo de validación se realiza un cálculo de error. El resultado final lo obtenemos a partir de realizar la media aritmética de los K valores de errores obtenidos, según la fórmula:

$$E = \frac{1}{K}\sum_{i=0}^{K} E_i$$

Ecuación 26. Error de la validación cruzada de K iteraciones.

Es decir, se realiza el sumatorio de los K valores de error y se divide entre el valor de K.

Error de la validación cruzada aleatoria

En la validación cruzada aleatoria a diferencia del método anterior, cogemos muestras al azar durante k iteraciones, aunque de igual manera, se realiza un cálculo de error para cada iteración. El resultado final también lo obtenemos a partir de realizar la media aritmética de los K valores de errores obtenidos, según la misma fórmula:

$$E = \frac{1}{K}\sum_{i=0}^{K} E_i$$

Ecuación 27. Error de la validación cruzada aleatoria.

Error de la validación cruzada dejando uno fuera

En la validación cruzada dejando uno fuera se realizan tantas iteraciones como muestras (N) tenga el conjunto de datos. De forma que para cada una de las N iteraciones se realiza un cálculo de error. El resultado final lo obtenemos realizando la media aritmética de los N valores de errores obtenidos, según la fórmula:

$$E = \frac{1}{N}\sum_{i=0}^{N} E_i$$

Ecuación 28. Error de la validación cruzada dejando uno fuera.

Donde se realiza el sumatorio de los N valores de error y se divide entre el valor de N.

La salida de la validación cruzada es una matriz de confusión basada en el uso de cada ejemplo etiquetado como un ejemplo de prueba exactamente una vez. Cuando un ejemplo es usado pata probar un clasificador, este no ha sido usado para entrenar el clasificador. De esta forma, la matriz de confusión obtenida por

la validación cruzada es intuitivamente un indicador justo del comportamiento de un algoritmo de aprendizaje en un conjunto de casos de prueba.

2.4.3 Curva ROC (acrónimo de Característica Operativa del Receptor)

"Característica Operativa del Receptor" (ROC por sus siglas en inglés) es una curva usada para resumir el comportamiento de un problema de decisión binario. En ella se representan la razón de verdaderos positivos (ejemplo, sensibilidad) contra la razón de falsos positivos del clasificador (Ver figura 6). El punto (0,0) en la gráfica describe un clasificador que sin falsos positivos pero tampoco detecta verdaderos positivos, mientras que el punto (1,1) representa un clasificador que solo puede asignar los datos a la clase positiva. Un clasificador que puede clasificar correctamente todos los ejemplos en el conjunto de entrenamiento es representado por el punto (0,1), mientras que un clasificador discreto que caiga en la línea discontinua de la figura de abajo genera solo clasificaciones aleatorias de los datos (71).

Los clasificadores que se encuentran por encima de la línea y cerca del extremo superior izquierdo generan pocos falsos positivos, mientras que los clasificadores que estén en el extremo superior derecho clasifican correctamente casi todos los positivos pero tienen una alta razón de falsos positivos.

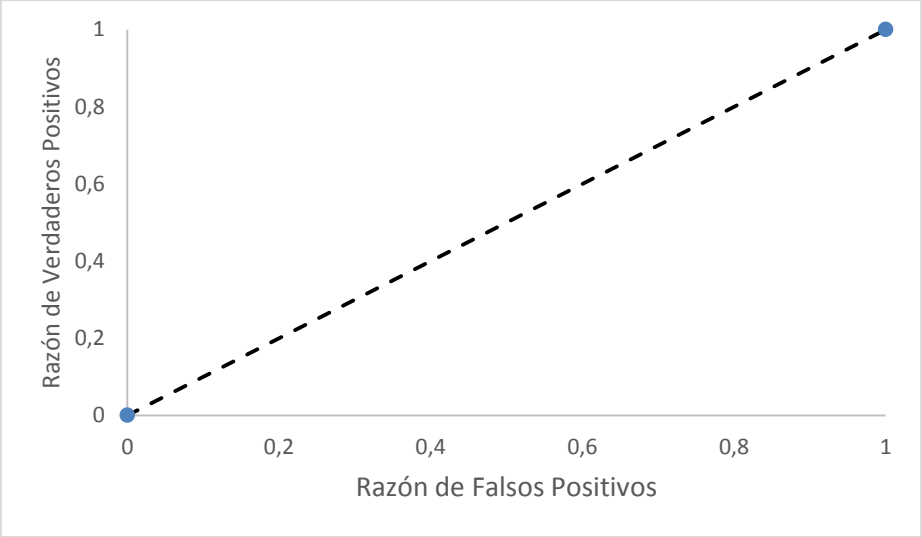

Figura 6. Característica Operativa del Receptor (ROC).

Las curvas ROC son utilizadas para caracterizar a los clasificadores que producen valores numéricos o puntuaciones para los ejemplos en el conjunto

de entrenamiento. Estas puntuaciones son utilizadas para clasificar los datos asignando un umbral a cada clase. Para un clasificador binario, los ejemplos son asignados a la clase positiva si la puntuación del ejemplo es mayor que el valor umbral de la clase. Cada umbral produce un punto diferente en una gráfica ROC y la correspondiente curva ROC puede ayudar a determinar el valor de umbral que debería ser seleccionado para optimizar la exactitud, minimizar el costo asociado a las clasificaciones incorrectas y/o la alguna combinación de ambos.

Para construir la curva ROC, los valores de *Razón de Falsos Positivos* y *Sensibilidad* son calculados para todos los posibles valores de corte que están contenidos en el rango de 0-1; la curva ROC es la representación gráfica de todos los falsos positivos contra su sensibilidad.

2.5 Conclusiones parciales

Las Máquinas de Vectores de Soporte, son una técnica de aprendizaje supervisado útil para resolver problemas de clasificación binaria; este es el método de clasificación propuesto por distintos autores en trabajos similares tratados anteriormente. La selección de atributos permite aumentar la exactitud del clasificador y al mismo tiempo reducir el costo computacional que presupone trabajar con casos cuya dimensión sobrepasa los 11000 atributos. Los métodos de filtrados y embebidos favorecen la reducción del espacio de representación de casos. Para la evaluación de los resultados las métricas de evaluación *Exactitud*, *Precisión*, *Sensibilidad* y *Especificidad* son empleadas por la validación cruzada en la realización de los ensayos con diferentes datos de entrenamiento y prueba.

Capítulo 3: Sistema para predecir interacciones entre dominios de proteínas

En el presente capítulo se detalla el proceso seguido para conformar la base de casos. Se introducen y calculan nuevos descriptores para pares de secuencias de aminoácidos, convirtiéndose en los atributos que describen los casos. Se seleccionan los rasgos más representativos aplicándose el clasificador SVM con el cual se alcanzan las mayores exactitudes. Por último, se diseña e implementa un software que permite la predicción de interacción de proteínas.

El algoritmo seguido en la modelación de la solución del problema es el siguiente:

1. Creación de la base de casos.
2. Selección de atributos.
3. Exploración de parámetros.
4. Selección del método de clasificación.
5. Evaluación de la propuesta.

A continuación se describen los distintos pasos.

3.1 Creación de la base de casos

La propuesta se enmarca en el trabajo con la estructura primaria de las proteínas, basándose únicamente en información obtenida de las secuencias de aminoácidos. La conformación de la base de casos utilizada se realizó con el empleo de los pares de dominios provenientes de las bases de datos 3did, Negatome e iPfam; descargadas el 23/9/2015, 5/10/2015 y 10/10/2015 respectivamente. Luego, cada par de dominios es caracterizado por un conjunto de descriptores numéricos para pares de secuencias obtenidos con ProtDCal y responde a una etiqueta clase: *Positiva***,** para aquellos pares de dominios con ocurrencia de interacción y *Negativa* para aquellos pares sin interacción. De esta forma un caso queda representado por el vector:

$$(Par\ de\ secuencias, Descriptor\ 1, Descriptor\ 2, \dots, Descriptor\ n, Clase)$$

Un ejemplo de caso es:

(PF02167 PF02921, 65.47, 37.4, 66.72,…..,47.68, 91.04, 17.84, 17.27, P)

donde:

PF02167: es el código de acceso al dominio cuya secuencia de aminoácidos es

HADIDVTDQAQIRRGLTVFTELCMGCHSAKYVTYRDLIDYPETSLSREDVDDL
RGDKPLIAGMVTDLAPEDAKVSYGKVPPDLSLIVSARRGGADYVYSILTGFEH
DPAGHVPDGNFNEYFPGNRIAMPDPLSWLGHDAADTADLEQQALDVSAFLAY
ISDPHQNERRAIGRYVVGFLILLTLVFYLLKKEIWKDI

PF02921: es el código de acceso al dominio cuya secuencia de aminoácidos es

HTDVKIPDFDPYRRNSLHDPNTETSSSDSERRAFSYLTVGSAGVATVYCSKYL
VETFISSMSASAD

P: Corresponde a la clase *Positiva*, lo cual indica que los dominios PF02167 y PF02921interactúan.

Los valores numéricos corresponden a los descriptores moleculares que describen numéricamente la interacción de ambas secuencias.

3.1.1 Selección de los pares de dominios para conformar la base de casos

Las bases de datos 3DID e iPfam contienen pares de dominios de proteínas que interactúan, aportando inicialmente 9326 y 9516 casos cada una, mientras que Negatome contribuyó con 2666 pares de dominios que no interactúan. Luego de una exploración inicial de los casos se detectó que en los repositorios de PPIs existían 7731 pares que se repetían en ambos conjuntos por lo que se eliminaron tales repeticiones, dejando la clase positiva con un total inicial de 11111 de casos, evidenciando desde el comienzo un desbalance entre el total de casos positivos y el total de casos negativos de aproximadamente 1/4.

Posteriormente se descubrió que existían 260 pares de dominios que se repetían en las datas de positivos y negativos. Puesto que Negatome está construido con la aplicación de varios filtros, donde uno de ellos es que un par en análisis no esté presente en 3did ni en iPFam; luego, la aparición de los pares repetidos puede deberse a la versión de la base de datos, ya que posiblemente 3did o iPFam hayan seguido creciendo y Negatome no haya sido

limpiado de nuevo. Por ende, esos casos son considerados "nuevos positivos" y deben eliminarse de los negativos, quedando ahora la clase negativa con 2406 casos.

Para el desarrollo de aplicaciones biomédicas es mucho más relevante el hecho de identificar interacciones entre proteínas o sus dominios que arribar a la ausencia de interacción entre estas estructuras. Es por ello que el número de casos que aparecen en las bases de datos de interacciones confirmadas supera considerablemente el volumen de casos almacenados en Negatome. Sin embargo, lo que naturalmente ocurre en estos sistemas es que la razón de desbalance converja a la no ocurrencia de interacción; por ejemplo, debido a que la cantidad de residuos en la interfaz de interacción entre dos proteínas o dominios es mucho menor que el número total de residuos en cada uno, razón por lo cual la combinatoria de pares de dominios sin interacción debe superar a los que muestran asociación. En este sentido, el número reducido de casos en Negatome es entendido como un limitado estudio de pares sin interacción. Como consecuencia para asegurar la consistencia y fiabilidad de los datos se decidió cruzar los casos negativos y positivos dejando solo aquellos pares de dominios cuyos dominios individuales estuviesen representados en Negatome, permitiendo de esta manera incrementar la calidad de los datos a la hora de representar cada clase del problema. Como resultado se obtuvo una disminución de la clase positiva a 1922 casos.

Los dominios que aparecen en cada uno de estos repositorios están representados por un código de acceso perteneciente a proteínas de Pfam, por lo que el paso siguiente fue obtener todas las secuencias simples de cada uno de los dominios en la base de datos de Pfam; la cual fue descargada el 12/10/2015. Una vez seleccionadas se observó que la mayor parte de los dominios contenían un alto número de secuencias. Para eliminar la redundancia de información por cada uno de los mismos se empleó CD-HIT (72); un programa diseñado para clusterizar y comparar grandes conjuntos de secuencias de proteínas.

En CD-HIT emplean un algoritmo de *clustering* incremental *greedy*. Las secuencias son ordenadas primeramente en orden decreciente de su longitud. La secuencia más larga se convierte en la más representativa del primer grupo.

Luego, cada secuencia a continuación es comparada con las más representativas de los grupos existentes, si la similitud con cualquiera de las representativas está por encima de un umbral dado, es agrupada en ese grupo; de otra manera, un nuevo grupo es definido con esa secuencia como la más representativa. El umbral de similitud puede ser definido por el usuario, para este estudio se llevó a cabo una reducción del 40 % (menor valor permisible definido por el programa) para desechar la mayor cantidad de secuencias posibles.

Aún después de la reducción un número grande de secuencias subsistieron por cada dominio, representar la interacción entre dos dominios significaba multiplicar la cantidad de secuencias de uno por la cantidad de secuencias del otro. Tal hecho conllevaba al crecimiento de la base de casos a un tamaño computacionalmente imposible de manejar. Para simplificar el problema se decidió entonces identificar, de las secuencias restantes por cada dominio, aquella que resultara ser la más representativa de todas para conformar la data final. Para ello se realizó primeramente por cada dominio un alineamiento de sus secuencias utilizando para ello Kalign3 (73); un programa para el alineamiento de secuencias de proteínas.

El alineamiento de las secuencias es el procedimiento de comparar dos o más secuencias mediante la búsqueda de una serie de patrones de caracteres individuales que se encuentran en el mismo orden en las secuencias (74). Dos secuencias son alineadas colocando ambas en una página en dos filas. Idénticos o similares caracteres son colocados en la misma columna, y caracteres diferentes pueden ser ubicados en la misma columna como una desigualdad o por el contrario se sitúa un hueco en la secuencia opuesta. Las secuencias que pueden ser fácilmente alineadas de este modo se dice que son similares (75).

Luego de que por cada dominio estuvieran alineadas todas sus secuencias, la búsqueda de la más representativa se realizó buscando aquella que estuviera más cercana al centro. Para ello primeramente se calculó por cada secuencia su similitud con cada una de las demás. Como resultado del alineamiento, por cada dominio, todas las secuencias quedaron iguales en longitud. Luego, la similitud de cada par se calculó sumando el número de veces en que en ambas

cadenas coincidían los aminoácidos en la misma posición; expresando finalmente el valor obtenido en porciento. Posteriormente por cada secuencia se sumaron las similitudes de la misma con el resto de las secuencias. La seleccionada como centroide fue aquella cuya suma alcanzase el máximo valor.

Para confirmar la validez del procedimiento de encontrar la secuencia más representativa de cada dominio para conformar el conjunto de casos se realizó un experimento que consistió en crear 10 datas, donde para cada dominio se selecciona la secuencia de forma aleatoria (Ver Anexo 3). Los resultados arrojados por el experimento muestran la factibilidad de seleccionar una secuencia por dominio de la forma descrita previamente.

Una vez identificada la secuencia más representativa de cada dominio se pasó a conformar el conjunto de casos final. Para predecir el comportamiento de un clasificador en datos nuevos, se requiere medir su taza de errores en un conjunto de ejemplos externos que no haya formado parte en la construcción del clasificador. Este conjunto de datos es denominado conjunto de prueba. Por tanto, una tarea de la clasificación normalmente involucra la separación de los datos en conjuntos de entrenamiento y prueba; donde ambos conjuntos deben ser ejemplos representativos del problema en cuestión (76).

El escenario más estricto para evaluar un clasificador y su capacidad de predicción es aquel en donde no existan casos para los cuales alguno de sus dominios esté repetido en los conjuntos de entrenamiento y predicción. Por consiguiente, la división de los casos en conjuntos de entrenamiento y prueba se realizó de forma que se considerase el criterio de que los pares de dominios en el entrenamiento y la prueba tuvieran o no un dominio en común.

Por tal motivo, para realizar las predicciones externas se decidió crear dos conjuntos de prueba, *Hard* y *Easy,* las cuales se describen a continuación:

- **Easy:** ambos dominios de un caso están representados en ejemplos tanto del conjunto de prueba como del entrenamiento. Además se añadió la restricción de que los dominios presentes en este conjunto solo aparecieran una vez. O sea, si en las pruebas entra una instancia AB es porque en el entrenamiento existe al menos un ejemplo con un

dominio A y al menos una con un dominio B y además en el conjunto de prueba no entrará ningún otro par que contenga A o B.

- **Hard:** al menos uno de los dos dominios de un ejemplo no está representado en el entrenamiento. Esto es si en el conjunto de prueba existe un caso AB, implica que en el entrenamiento no hay ningún caso que tenga al dominio A o al B.

Al realizar la división teniendo en cuenta las condiciones antes descritas, de un total de 4327 casos, prevalecieron para llevar a cabo el entrenamiento 3491 y para efectuar las pruebas 836; divididos en 426 para el conjunto *Easy* y 410 para el conjunto *Hard*. De esta forma quedaron divididos los datos en entrenamiento y prueba, donde se logra al menos que para las pruebas se separen el 20% de los casos y ambos subconjuntos (*Hard* y *Easy*) sean similares en cantidad. El conjunto de entrenamiento es también empleado para realizar validación interna por medio de validación cruzada de 10 pliegues.

Una vez bien definidos los pares de dominios a estudiar, se procedió a definir cuáles serían los atributos a aprender y sus valores.

3.1.2 Nuevos descriptores para pares de secuencias de aminoácidos

En el presente trabajo se introducen nuevos descriptores para pares de secuencias de aminoácidos, validando su aplicabilidad mediante la modelación de DDI. La adaptación de los descriptores para el cálculo de pares de secuencias se realizó definiendo los descriptores como el resultado del balance entre los descriptores de productos y reactivos de una reacción de dimerización (concatenación) entre dos cadenas de aminoácidos (A y B):

$$2A + 2B => AB + BA$$

Ecuación 29. Adaptación de descriptores pares de secuencias.

De esta forma un descriptor D(A-B) correspondiente al par A-B se calcularía como:

$$D(A-B) = D(AB) + D(BA) - 2D(A) - 2D(B)$$

Ecuación 30. Nuevos descriptores para pares de secuencias de aminoácidos.

Donde D(X) corresponde al valor del descriptor para una secuencia X. Siguiendo esta metodología se calcularon 13248 descriptores para cada par de dominios en todo el conjunto de datos.

3.2 Selección de atributos

La selección del conjunto de atributos se realiza en tres pasos:

1. Filtrado por Ganancia de la Información.
2. Filtrado por análisis de correlación.
3. Métodos Wrapper para seleccionar el subconjunto de atributos que mayor exactitud alcance.

Se empleó el software Weka (77)(78) para explorar este espacio de descriptores. Se recurrió al método de selección de atributos "InfoGainSubsetEval" para ordenar los descriptores de acuerdo a su capacidad para describir la distribución de las clases (pares positivos y negativos). Se seleccionaron aquellos atributos que mostraban una Ganancia de Información superior al 5% de la Información Total del conjunto de clases, de donde resultaron seleccionados 326 atributos.

Posteriormente, buscando eliminar aquellos atributos con un alto grado de similitud y así eliminar la redundancia en los datos, se utilizó un filtro que aplica un algoritmo de agrupamiento *single-linkage* (54) usando el coeficiente de correlación (cc) Spearman como métrica para el agrupamiento. En el algoritmo de clustering empleado se utiliza el coeficiente de correlación de Spearman como métrica para el agrupamiento. La suma del cc por cada miembro de un clúster se realiza para identificar el elemento más cercano al centro, que posteriormente es seleccionado como representante de todos los atributos del clúster en el conjunto de datos finalmente reducido. Se utilizó el valor 0.95 como umbral para el cc con la finalidad de eliminar solo aquellos que fuesen muy similares. Consecuentemente se redujo a 322 atributos la representación de cada caso, mostrando una diferencia poco significativa en comparación con la cantidad antes presentada.

Posterior a las técnicas de filtrado se aplicó, como técnica *Wrapper*, el método "WrapperSubsetEval" de Weka para seleccionar un subconjunto reducido de atributos con los cuales modelar el problema. Partiendo de los 322 atributos seleccionados con los filtros anteriores, empleando como técnica de modelación LibSVM (42), se seleccionaron diferentes subconjuntos de atributos variando la función kernel; sin modificar los valores de los parámetros que aparecen por defecto en la interfaz de Weka. El método de búsqueda utilizado

fue “GeneticSearch”, el cual realiza una búsqueda usando un algoritmo genético simple descrito por Goldberg en1989 (79).

Las funciones kernel empleadas fueron Polikernel y RBF, para las cuales ambas ejecuciones se resumen a continuación.

Selección de atributos con el kernel Polinomial

Se realizó una primera selección de atributos empleando el método de búsqueda “BestFirst” (80)(81) con dirección de búsqueda SFS. Posteriormente, se llevó a cabo una nueva selección de atributos pero esta vez con el método de búsqueda “GeneticSearch” sembrando en la población inicial un cromosoma con los atributos seleccionados con el método “BestFirst”. La población empleada por el algoritmo genético fue el espacio de descriptores, refiriéndose a ellos por su índice, donde el tamaño de la población fue igual a 20. Las probabilidades de cruce y de mutación fueron igual a 0.6 y 0.033 respectivamente.
Los valores de los parámetros de SVM utilizados para la selección fueron dejados tal cual aparecen por defecto en el LibSVM de Weka: el Costo igual a 1 y el grado del Polikernel (exponente) igual a 1. El desempeño de cada subconjunto fue evaluado en validación cruzada de 5 pliegues. Este análisis resultó en un subconjunto conformado por 19 atributos (Ver Anexo 1).

Seleccionados de atributos con el kernel RBF

De forma equivalente a la seguida con el kernel Polinomial se realizó una selección inicial aplicando el método de búsqueda “BestFirst”. Seguidamente se realizó una búsqueda con el método “GeneticSearch”.

Los valores de los parámetros de SVM utilizados para la selección fueron dejados tal cual aparecen por defecto en el LibSVM de Weka: el Costo igual a 1 y el parámetro Gamma del kernel RBF igual a 0.01. Como resultado se obtuvo un subconjunto de 16 atributos (Ver Anexo 2).

Con los datos seleccionados al aplicar estos filtros se procedió a realizar una exploración de parámetros, buscando evaluar la mejor configuración de parámetros y con cual función kernel se obtienen los mejores resultados en la modelación.

3.3 Exploración de parámetros

Con los subconjuntos obtenidos se realizó una exploración de parámetros para buscar aquella configuración de valores que aumentase la exactitud del clasificador. De forma paralela se realizaron exploraciones con ambos kernels (Polinomial y RBF) ajustando los parámetros requeridos por ambas variantes en la función LibSVM de Weka, evaluando los resultados en validación cruzada de 10 pliegues.

Kernel Polinomial

Los parámetros a variar cuando se emplea el kernel polinómico son el Costo y el Exponente. Un conjunto de modelos fueron generados variando cada uno de estos parámetros de la siguiente forma:

El costo estuvo enmarcado en el intervalo $[2^{-5}; 2^{9}]$ con paso 14, lo que significa ir incrementando en cada modelación el valor del exponente en 1 al ser:

$$Incremento = \frac{CostoMax - CostoMin}{CostoPaso}$$

Ecuación 31. Cálculo del incremento.

Donde inicialmente $CostoMax = 9$, $CostoMin = -5$ y $CostoPaso = 14$

De esta forma fueron generados 15 valores de costo distintos turnados desde de 0,03125 hasta 512, permitiendo ampliar el rango de valores a probar. Todos los valores de costo fueron probados con los exponentes del polinomio hasta el grado 3.

Kernel RBF

Los parámetros a ajustar con el Kernel RBF son el Costo y el parámetro Ganma. Ambos parámetros fueron variados de forma equivalente a como se realizó con el kernel Polinomial. El parámetro Ganma estuvo enmarcado en el intervalo $[2^{-15}; 2^{3}]$ con paso 18.

A continuación se muestra una tabla resumen con los valores de los parámetros para los cuales se obtuvieron los modelos con la exactitud más alta.

Tabla 5. Resumen de exploración de parámetros.

Polikernel			**RBF**		
Parámetros	**CV**	**TS**	**Parámetros**	**CV**	**TS**
C=2 y E=1	69.92	70	C=4 y G=0.125	69.87	71.1

3.4 Selección del método de clasificación

A continuación, se muestra un esquema radial (Ver Figuras 7 y 8) donde están representadas las métricas: exactitud, sensibilidad, precisión y especificidad para cada modelo construido durante de la exploración de los parámetros.

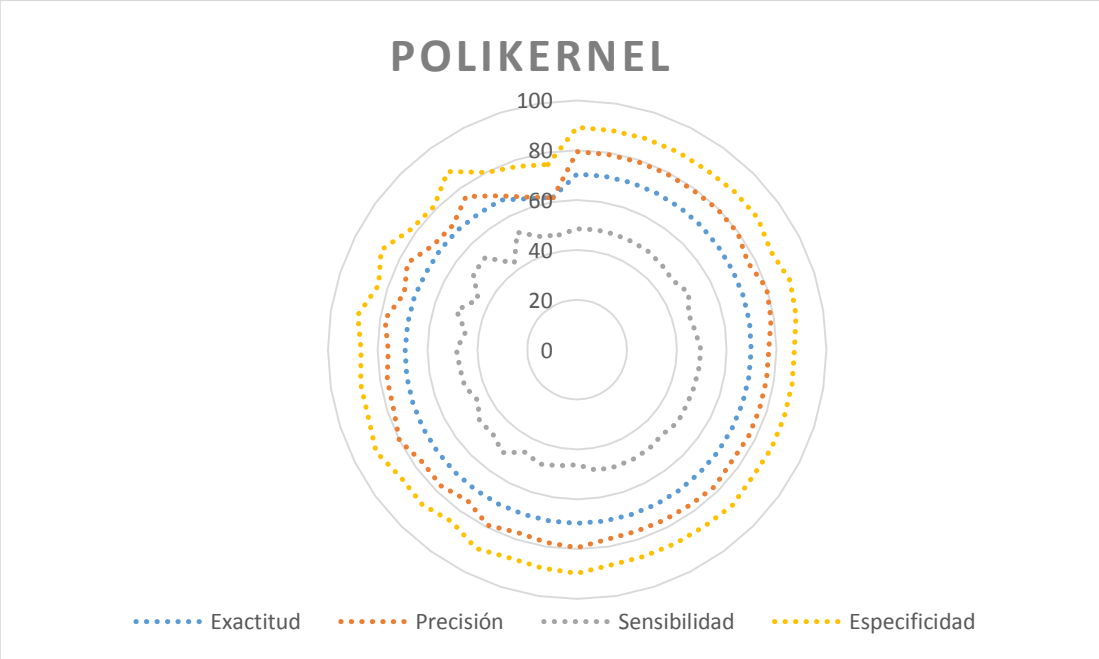

Figura 7. Exploración de parámetros de SVM con Polikernel.

Los puntos, de cada serie de datos, ubicados a lo largo de la misma directriz radial corresponden a un mismo modelo. Los modelos aparecen ordenados en sentido de las manecillas del reloj en orden decreciente de la Exactitud. Las circunferencias delimitan los valores porcentuales de cada métrica. Del análisis del gráfico, se observa como con el kernel polinomial se generan modelos uniformes en los valores de las métricas analizadas (poco sensibles a cambios en el costo de la maquina o en el coeficiente de polinomio), en donde la Exactitud en todos los modelos se mantiene entre el 69 y 70%.

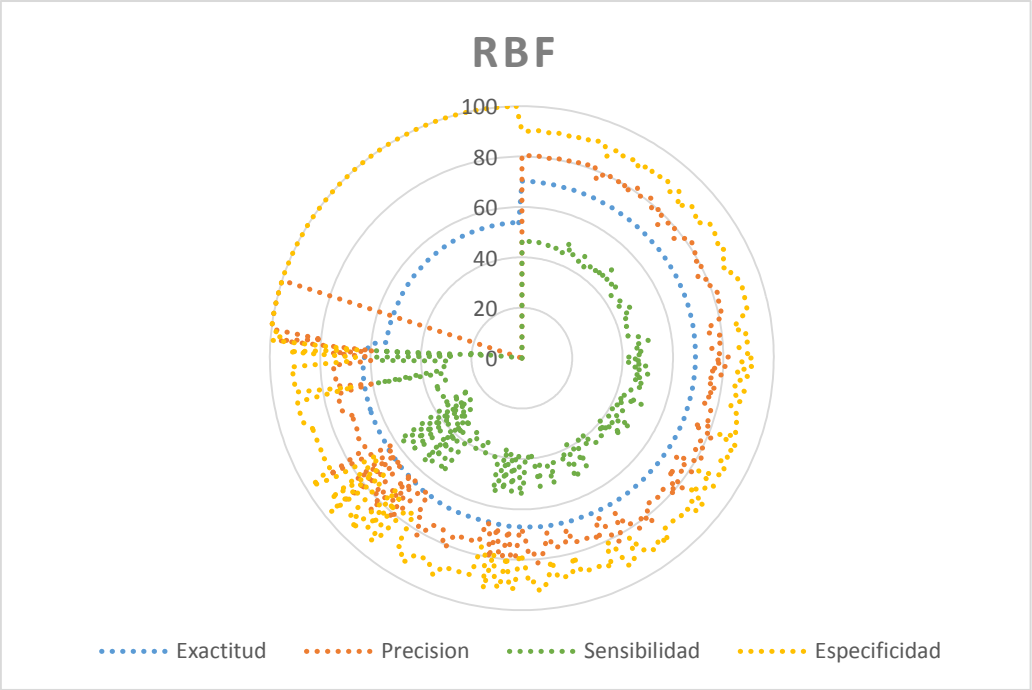

Figura 8. Exploración de parámetros de SVM con RBF.

Con el kernel RBF se observa mayor diversidad en el desempeño de los modelos según los valores de los parámetros adoptados en cada uno. Se observan varios modelos donde la precisión alcanza el 100 % mientras que la sensibilidad es igual a 0, con una Exactitud que no sobrepasa el 55%. Sin embargo destacan varios modelos con métricas muy similares a las alcanzadas con el kernel Polinomial, donde la más alta alcanza el 70,24% para valores de precisión y especificidad superiores a 80% y 90% respectivamente.

A pesar de que las exactitudes obtenidas por el modelo seleccionado utilizando las diferentes funciones kernel son casi iguales, en el caso del polikernel las mayores exactitudes se obtienen con el grado del polinomio igual a 1, lo cual disminuye la complejidad del modelo al estar en presencia de un problema donde las clases son separables linealmente en el espacio de rasgos originales. Por tales motivos se selecciona el clasificador creado con la variante del kernel Polinomial para mostrar los resultados de la investigación.

3.5 Evaluación de la propuesta

El método de evaluación empleado fue la curva ROC, la cual es un instrumento para visualizar y seleccionar clasificadores basándose en la relación existente entre los beneficios (verdaderos positivos) y el costo (falsos positivos) (82).

Una vez obtenido el modelo se prosiguió a evaluar su desempeño al clasificar los casos del conjunto de prueba conformado al inicio. Este conjunto se subdividía en dos subconjuntos: *Easy* y *Hard*. Con el objetivo de comparar sus desempeños se evaluó el modelo para los tres casos: *Easy*, *Hard* y el conjunto resultante de la unión de ambos.

La evaluación se realizó mostrando para cada caso la distribución de probabilidades de cada clase. Posteriormente se seleccionaron las probabilidades correspondientes a la clase *Positiva* ordenadas en orden decreciente. Luego, se calcularon las diferentes métricas de evaluación, eligiendo cada valor de probabilidad como umbral de decisión para clasificar cada caso entre la clase *Positiva* y *Negativa*. En la Figura 9 se muestran, para cada conjunto de prueba, los valores de Sensibilidad contra la Razón de Falsos Positivos por cada umbral de decisión escogido.

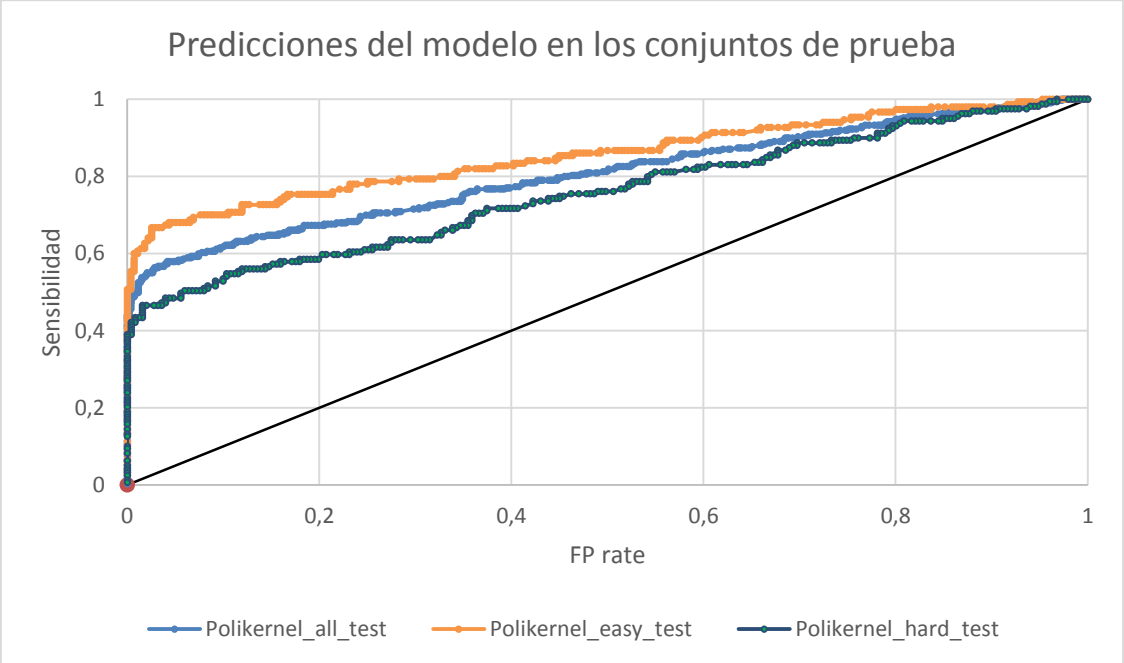

Figura 9. Curva ROC de predicciones del modelo en los conjuntos de prueba.

En la figura se observa cómo se generan modelos para los cuales a una razón de falsos positivos muy baja la sensibilidad aumenta.

Comparación del clasificador con predictores externos

El resultado del trabajo solo puede ser valorado si ha sido comparado con los resultados arrojados por predictores externos. Para este estudio se emplean tres de los métodos computacionales más relevantes en el marco de este problema: PIPE, Pre-PPI y SPPS.

Las datas de prueba *Easy* y *Hard* en conjunto fueron empleadas en corridas realizadas en cada uno de los servidores externos, los cuales arrojaron una probabilidad de interacción por cada par de secuencias. Luego, siguiendo un proceso equivalente al anterior, se calcularon las métricas de evaluación para cada uno de los métodos y se conformaron las Curvas ROC con tales resultados. (Ver Figura 10).

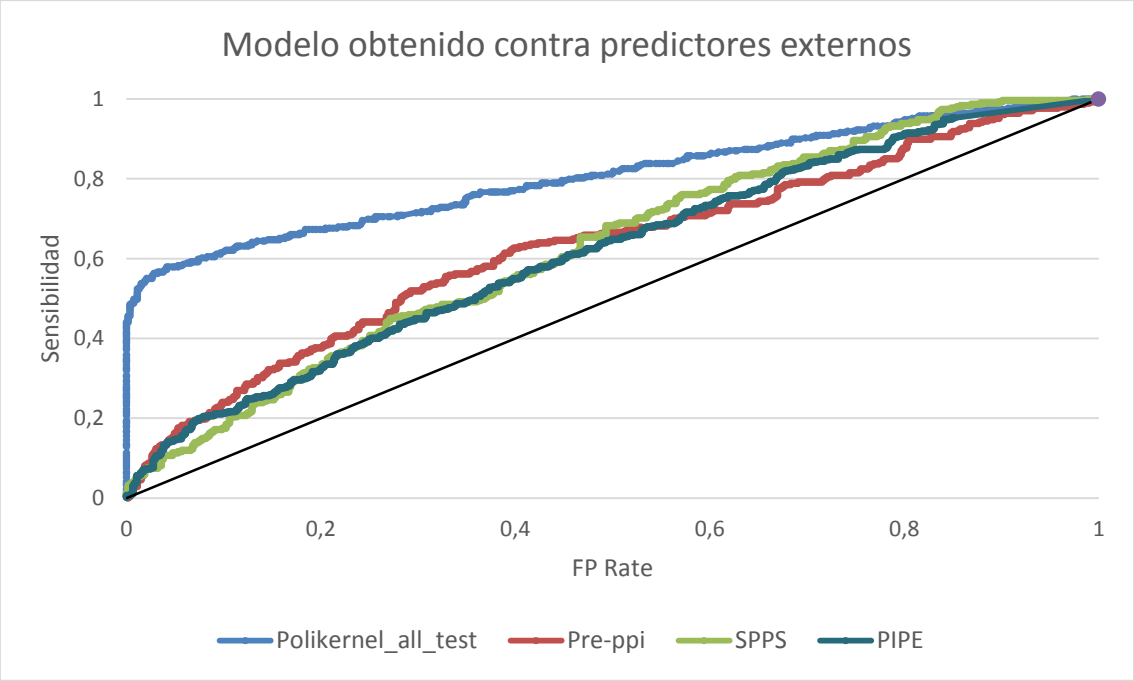

Figura 10. Curvas ROC del modelo obtenido contra predictores externos.

En la Figura se observa como el clasificador obtenido supera notablemente a los clasificadores con los cuales se compara, al ser el más cerca se encuentra del extremo superior izquierdo, generando muy pocos falsos positivos. No obstante es necesario tener en cuenta que los predictores externos fueron construidos para clasificar interacción entre pares de proteínas, lo cual constituye un sesgo, al ser el modelo generado un clasificador de interacciones entre dominios de proteínas, ya que las secuencias de proteínas con

interacción pueden contener partes que no interactúen, no ocurriendo igual con las secuencias de dominios.

3.6 Diseño e Implementación del software

Una vez obtenido el modelo de clasificación se hizo necesario desarrollar un software amigable y multiplataforma para facilitar su utilización por la comunidad científica.

El software permite:

- ✓ Calcular descriptores numéricos con ProtDCal.
- ✓ Emplear las clases de Weka para utilizar el modelo.

Por tal motivo algunas de las funcionalidades de estos sistemas debían ser empleadas en la secuencia de actividades del nuevo sistema en desarrollo. Es por ello que se decidió desarrollar el software en lenguaje Java, el cual es un lenguaje de programación de propósito general, orientado a objetos y que permite la ejecución de un mismo programa en múltiples sistemas operativos.

3.6.1 Requerimientos de software

Los requerimientos de software son las condiciones o capacidades que el sistema debe cumplir para llegar a un entendimiento sobre lo que debe y lo que no debe hacer el mismo; a continuación se listan los requisitos funcionales y los requisitos no funcionales del sistema (83).

Requisitos funcionales:

- ✓ Predecir la clase a la que pertenecen nuevos casos empleando el modelo seleccionado.
    - ✓ Calcular descriptores con ProtDCal.
    - ✓ Clasificar nuevos ejemplos.
- ✓ Modificar la base de casos.
    - ✓ Calcular descriptores con ProtDCal.
    - ✓ Entrenar el clasificador.

- ✓ Evaluar el modelo de clasificación.

Requisitos no funcionales:

Los requerimientos no funcionales, son las propiedades o cualidades que el producto debe tener, es decir, las características del producto a desarrollar.

Usabilidad:

- ✓ Para utilizar el programa es necesario que se encuentre instalada en el Sistema Operativo el JRE (del inglés *Java Runtime Enviroment*), versiones 1.7 o superior.

3.6.2 Diseño del sistema

Luego en la etapa de diseño se identificaron los actores y casos de uso necesarios para satisfacer los requisitos mencionados (Ver figura 11). Para ello se tuvieron en cuenta los sistemas ProtDCal y Weka, al ser necesario invocar algunas de sus funcionalidades en la ejecución del nuevo software.

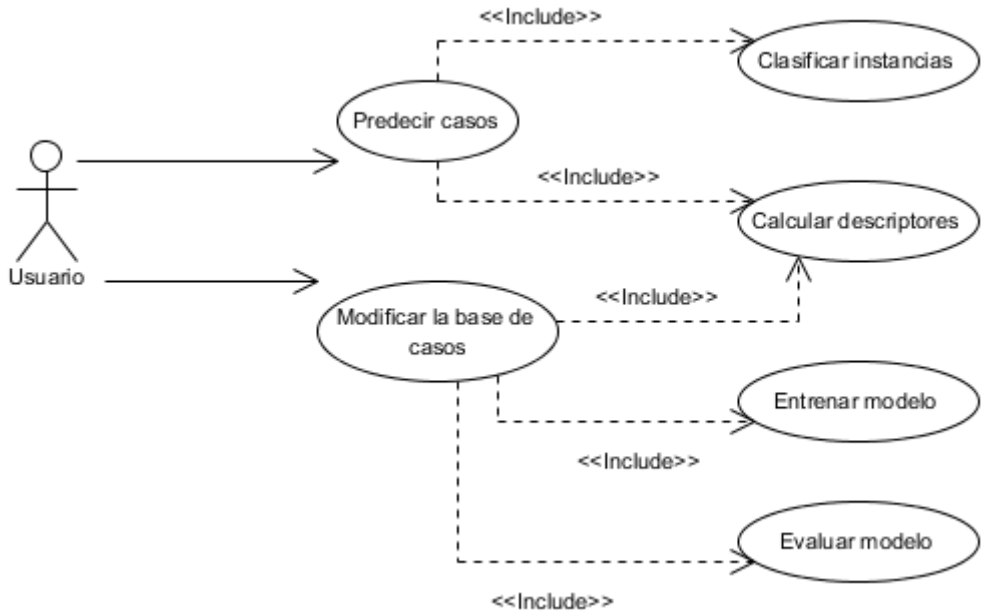

Figura 11. Diagrama de casos de uso.

Posteriormente un diagrama de componentes fue modelado para representar cómo estaría el sistema dividido en componentes, así como mostrar las dependencias entre ellos (Ver figura 12). El sistema cuenta con los componentes fundamentales:

***Prediction*:** contiene las funcionalidades desarrolladas para predecir la clase a la que pertenecen nuevos casos empleando el modelo seleccionado.

***ReTrain*:** contiene las funcionalidades desarrolladas para modificar la base de casos.

weka.jar: contiene las clases de *weka* a emplear para satisfacer los requisitos funcionales, los métodos principales pertenecen a la clase *Classifier* y son:

- ✓ *buildClassifier:* se encarga de la construcción del modelo del clasificador tomando como parámetro los ejemplos de entrenamiento; ejecutado por el componente *ReTrain*.
- ✓ *classifyInstance:* permite clasificar un ejemplo concreto. Devuelve la clase en la que se ha clasificado o un valor perdido si no se consigue clasificar; ejecutado por el componente *Prediction*.
- ✓ *distributionForInstance:* cumple la misma función que el método anterior, solo que el resultado de la clasificación se devuelve en forma de vector, con el grado de pertenencia del ejemplo dado a cada una de las clases; ejecutado por el componente *Prediction*.

***LibSVM.jar*:** contiene la clase LibSVM, que permite crear una nueva Máquina de Vectores de Soporte, permitiendo cambiar sus atributos. Para ello emplea funcionalidades implementadas en clases pertenecientes al componente *libsvm_class.jar.*

polik.model: corresponde al modelo de clasificación del sistema.

Descriptors: contiene las funcionalidades que permiten el cálculo de los descriptores numéricos con ProtDCal, para lo que cual se hacen referencias a los componentes *ProtDCal.jar* y *ppi_project.idl*.

ProtDCal.jar: contiene las funcionalidades de ProtDCal agrupadas en la clase *PPISpecificTask* que a su vez hereda de la clase *SpecificTask*, para el cálculo de descriptores para pares de secuencias. Para ello necesita del componente *ppi_project.idl.*

ppi_project.idl: contiene los nombres de los descriptores a calcular.

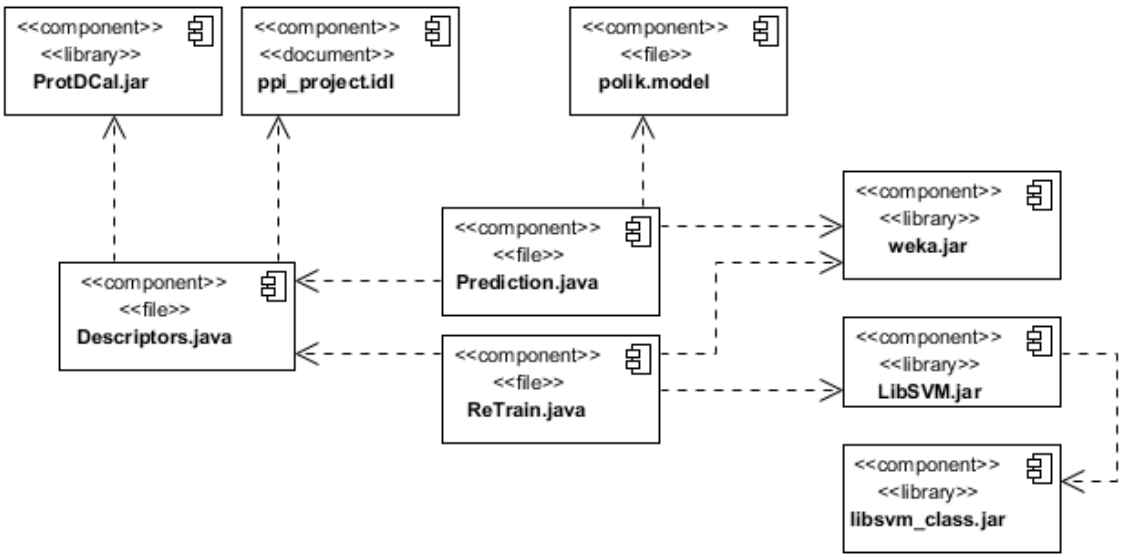

Figura 12. Diagrama de componentes del sistema.

Por último, en la etapa de diseño se realizó un diagrama de clases por cada requisito con el objetivo de definir las clases necesarias en la solución del problema. Dentro del conjunto de clases se incluyen aquellas correspondientes a los softwares *Weka* y *ProtDCal* a emplearse en el flujo del programa. La figura 13 corresponde al Diagrama de clases involucradas en la predicción, para visualizar el Diagrama de clases que comprenden el reentrenamiento del modelo ver Anexo 4.

Con la visualización de los diagramas de clases creados se logra una mejor comprensión del problema y de la solución que se propone.

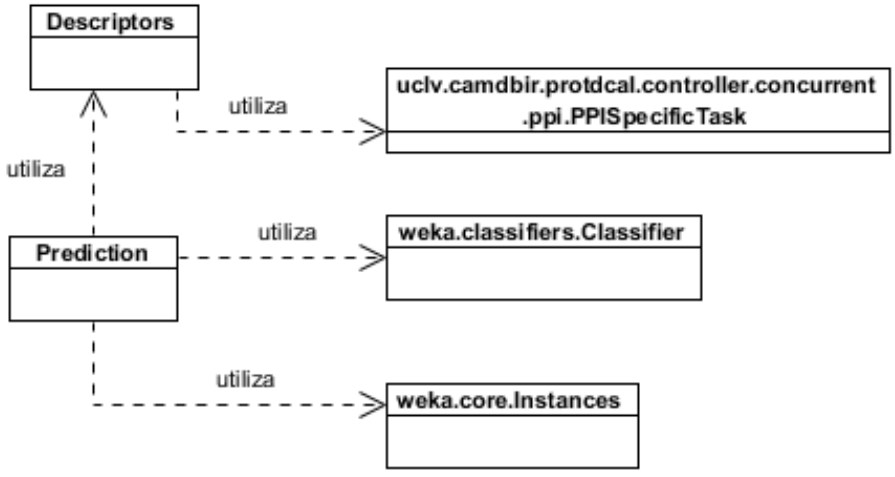

Figura 13. Diagrama de clases involucradas en la predicción.

3.6.3 Manual de usuario

El espacio de trabajo de PredDDI son las carpetas del programa:

- ✓ Datasets: Contiene todas las bases de datos usadas por el modelo en formato ARFF.
- ✓ Outputs: Contiene los archivos de salida (con los descriptores) del programa (* arff, *txt).
- ✓ Projects: Contiene el archivo de proyecto usado por el programa (*.idl).

Cuando la aplicación es ejecutada se muestra la siguiente interfaz:

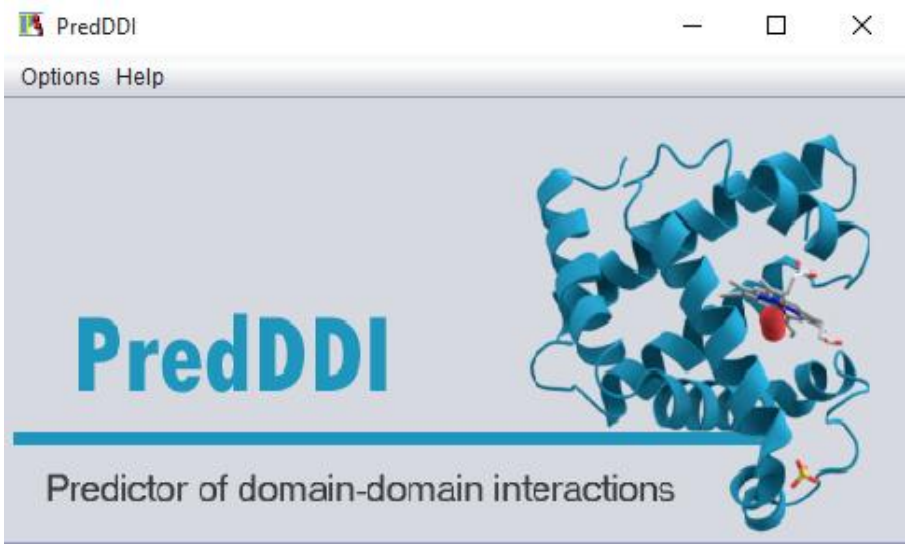

Figura 14. Ventana principal del software.

Desplegando el menú “Opciones” es posible acceder a las siguientes funciones:

1. **Execute predictions. (Ejecutar predicciones)**

Esta opción es para predecir posibles interacciones para casos no vistos con el modelo del programa.

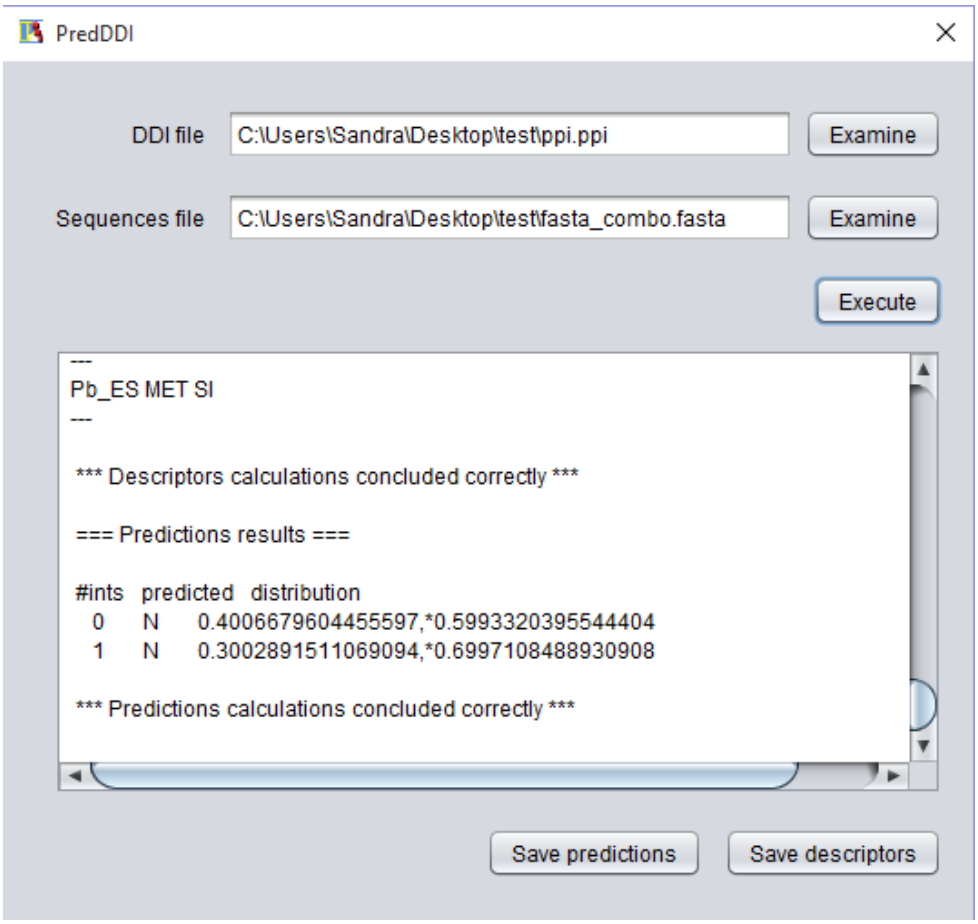

Figura 15. Ventana para efectuar predicciones.

Para ejecutar esta funcionalidad se necesita un archivo que contenga todas los pares de interacciones a ser procesados separados por “;” y con extensión ".ppi". Un ejemplo podría ser "ppi.ppi" con las líneas:

PA;PB

PA;PC

Y además necesita un archivo que contenga las secuencias *PA*,*PB* y *PC* en formato FASTA:

>PA
...SECUENCIA DE PA...
>PB
....SECUENCIA DE PB...
>PC
....SECUENCIA DE PC...

Una vez hayan sido seleccionados estos archivos, dando un clic en el botón "Ejecutar" se realizan los cálculos y la salida con los resultados de la predicción es mostrada en el área de texto de abajo.

Luego aparecen dos opciones:

- ✓ *Salvar la predicción:* Para salvar el resultado en una ubicación seleccionada por el usuario.
- ✓ *Salvar los descriptores:* Para salvar los descriptores calculados con ProtDCal en una ubicación seleccionada por el usuario.

2. Retrain the model. (Reentrenar el modelo)

Esta opción permite añadir nuevos casos a la base de casos con la que fue entrenado el modelo del programa para volverlo a entrenar.

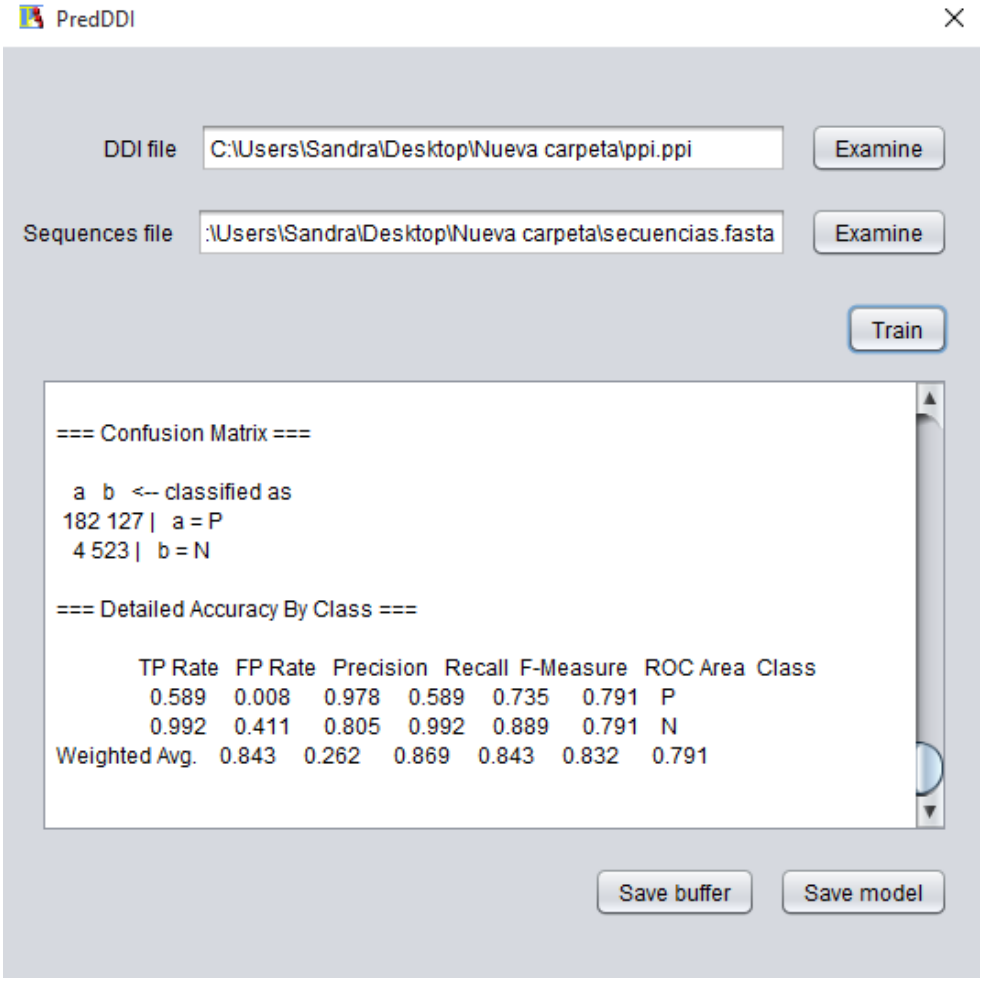

Figura 16. Ventana para entrenar el modelo nuevamente.

Para reentrenar el modelo se necesita un archivo que contenga los pares con interacción reportada a añadirse separados por ";" y extensión ".ppi". Se

necesita además por cada par la especificación de la clase a la cual pertenece. Un ejemplo podría ser el archivo "ppi.ppi" con las líneas:

PA;PB:P

PA;PC;N

Y además necesita un archivo que contenga las secuencias *PA*, *PB* y *PC* en formato FASTA:

>PA

...SECUENCIA DE PA...

>PB

....SECUENCIA DE PB...

>PC

....SECUENCIA DE PC...

Una vez hayan sido seleccionados estos archivos, dando un clic en el botón “Entrenar” se realizan los cálculos y la salida con los resultados del entrenamiento es mostrada en el área de texto de abajo.

Luego aparecen dos opciones:

- *Salvar el buffer:* Para salvar información útil acerca del modelo tal como la Precisión, Sensibilidad, etc.
- *Salvar el modelo:* Para salvar el modelo construido.

3.7 Conclusiones parciales

Con una base de casos conformada por vectores de descriptores numéricos obtenidos con ProtDCal se logró aprender un modelo de clasificación para identificar interacciones entre dominios de proteínas cuyo desempeño supera el desempeño de otros tres métodos computacionales, donde en dos de ellos se emplean también descriptores numéricos y SVM.

Los pasos seguidos en la creación de la base de casos, la adecuada selección de los atributos más representativos y el ajuste de parámetros realizado permitieron obtener modelos SVM con los kernel Polinómico y RBF con una alta sensibilidad y una baja razón de falsos positivos.

Por último, se desarrolló un software en Java que permite predecir la interacción entre dominios de proteínas desde una interfaz multiplataforma y amigable.

Conclusiones

- ✓ Se crea la base de casos de DDI constituida por 4326 casos descritos por 13246 atributos de alta fiabilidad, que queda disponible como conjunto standard de pruebas para futuros estudios en la temática.
- ✓ Se introducen nuevos descriptores numéricos libres de alineamiento, por el grupo de investigación de Descubrimiento de Fármacos y Bioinformática, para pares de secuencias de aminoácidos, a partir de los descriptores introducidos en el programa ProtDCal.
- ✓ A partir de la aplicación de diferentes métodos de selección de rasgos se reduce la dimensión de la base de casos a 19 atributos que garantizan mejoras en la precisión de los resultados.
- ✓ Se implementa la variante de Polikernel para SVM, la cual reporta mejores resultados que la variante RBF, con el ajuste de los parámetros correspondientes y soportada por una validación cruzada de 10 pliegues.
- ✓ El clasificador obtenido supera notablemente a los métodos de predicción de PPI más relevantes empleando igualmente la técnica SVM, lo cual valida la aplicabilidad de los nuevos descriptores numéricos definidos para pares de proteínas.
- ✓ Se implementa PredDDI que permite la predicción de la ocurrencia de interacción entre pares de dominios, con una interfaz amigable y portable. Las funcionalidades desarrolladas permiten a científicos realizar modificaciones a la base de casos.

Recomendaciones

- ✓ Extender el estudio realizado sobre la interacción entre dominios de proteínas a la predicción de interacciones entre pares de proteínas con sus secuencias completas.

Referencias bibliográficas

1. ASHBURNER, Michael. Gene Ontology: tool for the unification of biology. *Nature genetics*. 2000. Vol. 25, no. 1, p. 25-29. DOI 10.1038/75556.

2. NELSON, David L. y COX, Michael M. *Principles of Biochemistry*. 4th. New York : W.H. Freeman and Company, 2005. ISBN 0-7167-4339-6.

3. TORRE, Victor de la. Interacciones proteína-proteína: bases de datos y métodos teóricos de predicción. *Biotecnología Aplicada*. 2003. Vol. 20, no. 3, p. 201-208.

4. CRICK, Francis. Central Dogma of Molecular Biology. *Nature*. 1970. Vol. 227, p. 561-563.

5. KORASICK, David y JEZ, Joseph. Protein Domains : Structure , Function , and Methods. *Washington University in St. Louis, USA*. 2016. DOI 10.1016/B978-0-12-394447-4.10011-2.

6. MOSCA, Roberto, CÉOL, Arnaud, STEIN, Amelie, OLIVELLA, Roger y ALOY, Patrick. 3did: A catalog of domain-based interactions of known three-dimensional structure. *Nucleic Acids Research*. 2014. Vol. 42, no. D1, p. 1-6. DOI 10.1093/nar/gkt887.

7. FINN, Robert D, MILLER, Benjamin L, CLEMENTS, Jody y BATEMAN, Alex. iPfam : a database of protein family and domain interactions found in the Protein Data Bank. *Nucleic Acids Research*. 2014. Vol. 42, no. December 2013, p. 364-373. DOI 10.1093/nar/gkt1210.

8. SMIALOWSKI, Pawel, PAGEL, Philipp, WONG, Philip, BRAUNER, Barbara, DUNGER, Irmtraud, FOBO, Gisela, FRISHMAN, Goar, MONTRONE, Corinna, RATTEI, Thomas, FRISHMAN, Dmitrij y RUEPP, Andreas. The Negatome database: a reference set of non-interacting protein pairs. *Nucleic acids research*. 2010. Vol. 38. DOI 10.1093/nar/gkp1026.

9. PITRE, Sylvain. *PIPE : A protein- protein interaction prediction engine based on the re-occurring short polypeptide sequences between known interacting protein pairs*. Carleton university. Ontario. Ottawa, 2010.

10. THANGUDU, Ratna Rajesh, BRYANT, Stephen H, PANCHENKO, Anna R y MADEJ, Thomas. Modulating Protein - Protein Interactions with Small Molecules : The Importance of Binding Hotspots. *Journal of Molecular Biology*. 2012. Vol. 415, no. 2, p. 443-453. DOI 10.1016/j.jmb.2011.12.026.

11. PUNTA, Marco, COGGILL, Penny C, EBERHARDT, Ruth Y, MISTRY, Jaina, TATE, John, BOURSNELL, Chris, PANG, Ningze, FORSLUND, Kristoffer, CERIC, Goran, CLEMENTS, Jody, HEGER, Andreas, HOLM, Liisa, SONNHAMMER, Erik L L, EDDY, Sean R, BATEMAN, Alex y FINN, Robert D. The Pfam protein families database. *Nucleic Acids Research*. 2012. Vol. 40, no. November 2011, p. 290-301. DOI 10.1093/nar/gkr1065.

12. REIMAND, Jüri, HUI, Shirley, JAIN, Shobhit, LAW, Brian y BADER, Gary D. Domain-mediated protein interaction prediction: From genome to network. *FEBSLetters*. 2012. Vol. 586, p. 2751-2763. DOI 10.1016/j.febslet.2012.04.027.

13. O´MALLEY, Christopher J, MONTAGUE, Gary A, MARTIN, Elaine B, LIDDELL, John M, KARA, Bo y TITCHENER-HOOKER, Nigel J. Utilisation of key descriptors from protein sequence data to aid bioprocess route selection. *Food and Bioproducts Processing*. 2012. Vol. 90, no. 4, p. 755-761. DOI 10.1016/j.fbp.2012.01.005.

14. LI, Z. R., LIN, H. H., HAN, L. Y., JIANG, L., CHEN, X. y CHEN, Y. Z. PROFEAT: a web server

for computing structural and physicochemical features of proteins and peptides from amino acid sequence. *Nucleic Acids Research* [online]. 2006. Vol. 34, no. Web Server, p. W32-W37. DOI 10.1093/nar/gkl305. Available from: http://nar.oxfordjournals.org/lookup/doi/10.1093/nar/gkl305

15. RAO, H. B., ZHU, F., YANG, G. B., LI, Z. R. y CHEN, Y. Z. Update of PROFEAT: a web server for computing structural and physicochemical features of proteins and peptides from amino acid sequence. *Nucleic Acids Research* [online]. 2011. Vol. 39, no. Web Server, p. W385-W390. DOI 10.1093/nar/gkr284. Available from: http://nar.oxfordjournals.org/lookup/doi/10.1093/nar/gkr284

16. RUIZ, Yasser B, PAZ, Waldo, GREEN, James y MARRERO, Yovani. ProtDCal: A program to compute general-purpose-numerical descriptors for sequences and 3D-structures of proteins. *BMC Bioinformatics* [online]. 2015. Vol. 16, no. 1, p. 1-15. DOI 10.1186/s12859-015-0586-0. Available from: http://www.biomedcentral.com/1471-2105/16/162

17. SHEN, Hong-Bin y CHOU, Kuo-Chen. PseAAC: a flexible web server for generating various kinds of protein pseudo amino acid composition. *Analytical Biochemistry*. 2008. Vol. 373, no. 2, p. 386–388.

18. TODESCHINI, Roberto y CONSONNI, Viviana. Handbook of Molecular Descriptors. *New York*. 2000. Vol. 11, p. 688. DOI 10.1002/9783527613106.

19. PARK, Yungki. Critical assessment of sequence-based protein-protein interaction prediction methods that do not require homologous protein sequences. *BMC Bioinformatics* [online]. 2009. Vol. 10, no. 1, p. 419. DOI 10.1186/1471-2105-10-419. Available from: http://www.biomedcentral.com/1471-2105/10/419

20. PITRE, Sylvain, DEHNE, Frank, CHAN, Albert, CHEETHAM, Jim, DUONG, Alex, EMILI, Andrew, GEBBIA, Marinella, GREENBLATT, Jack, JESSULAT, Mathew, KROGAN, Nevan, LUO, Xuemei y GOLSHANI, Ashkan. PIPE : a protein-protein interaction prediction engine based on the re-occurring short polypeptide sequences between known interacting protein pairs. *BMC Bioinformatics*. 2006. Vol. 15, p. 1-15. DOI 10.1186/1471-2105-7-365.

21. LIU, Xinyi, LIU, Bin, HUANG, Zhimin, SHI, Ting, CHEN, Yingyi y ZHANG, Jian. SPPS : A Sequence-Based Method for Predicting Probability of Protein-Protein Interaction Partners. . 2012. Vol. 7, no. 1, p. 1-6. DOI 10.1371/journal.pone.0030938.

22. GUO, Yanzhi, YU, Lezheng, WEN, Zhining y LI, Menglong. Using support vector machine combined with auto covariance to predict protein – protein interactions from protein sequences. . 2008. Vol. 36, no. 9, p. 3025-3030. DOI 10.1093/nar/gkn159.

23. GUILLÉN, M Victoria. Estructura y propiedades de las proteínas. .

24. OUELLETTE, Robert J. Amino Acids, Peptides, and Proteins. In : *Principles of Organic Chemistry*. 2015. p. 371–396.

25. GARRETT, Reginald H. y GRISHAM, Charles M. Chemistry Is the Logic of Biological Phenomena. In : *Molecular Components of Cells. Part I*. [no date].

26. DAMODARAN, Srinivasan. *Aminoácidos, péptidos y proteínas*. University of Wisconsin-Madison. Madison. Wisconsin.

27. BERG, Jeremy M., TYMOCZKO, John L. y STRYER, Lubert. *Biochemistry*. 6. New York : W.H. Freeman and Company. New York, 2007. ISBN 0-7167-8724-5.

28. MARTÍNEZ, Juan José. *Libro electrónico de Bioquímica*. 2014. ISBN 978-607-8359-26-4.

29. ISAAC, Arnold E. y ARUMUGAM, Amala. Protein contact maps : A binary depiction of protein 3D structures. *Physica A* [online]. 2016. DOI 10.1016/j.physa.2016.08.033. Available from: http://dx.doi.org/10.1016/j.physa.2016.08.033

30. FEDUCHI, Elena, ROMERO, Carlos, YÁÑEZ, Esther, BLANCO, Isabel y GARCÍA-HOZ, Carlota. Bioquímica: conceptos esenciales. In : . Panamericana, 2011. ISBN 978-84-9835-357-0.

31. KELLEY, Lawrence A y STERNBERG, Michael J E. Protein structure prediction on the Web : a case study using the Phyre server. *Nature*. 2009. Vol. 4, no. 3. DOI 10.1038/nprot.2009.2.

32. HEGER, Andreas y HOLM, Liisa. Exhaustive Enumeration of Protein Domain Families. . 2003. Vol. 2836, no. 03, p. 749-767. DOI 10.1016/S0022-2836(03)00269-9.

33. What is Bioinformatics? *European Bioinformatics Institute*. 2006.

34. SCHOENROCK, Andrew, BURNSIDE, Daniel, WONG, Alex y DEHNE, Frank. Engineering Inhibitory Proteins with InSiPS : The In-Silico Protein Synthesizer. . DOI http://dx.doi.org/10.1145/2807591.2807630.

35. BERMAN, Helen, HENRICK, Kim, NAKAMURA, Haruki y MARKLEY, John L. The worldwide Protein Data Bank (wwPDB): ensuring a single , uniform archive of PDB data. *Nucleic Acids Research*. 2007. Vol. 35, no. November 2006, p. 2006-2008. DOI 10.1093/nar/gkl971.

36. KERRIEN, Samuel, ARANDA, Bruno, BREUZA, Lionel, BRIDGE, Alan, BROACKES-CARTER, Fiona, CHEN, Carol, DUESBURY, Margaret, DUMOUSSEAU, Marine, FEUERMANN, Marc, HINZ, Ursula, JANDRASITS, Christine, JIMENEZ, Rafael C, KHADAKE, Jyoti, MAHADEVAN, Usha, MASSON, Patrick, PEDRUZZI, Ivo, PFEIFFENBERGER, Eric, PORRAS, Pablo, RAGHUNATH, Arathi, ROECHERT, Bernd, ORCHARD, Sandra y HERMJAKOB, Henning. The IntAct molecular interaction database in 2012. . 2012. Vol. 40, no. November 2011, p. 841-846. DOI 10.1093/nar/gkr1088.

37. BEN-HUR y NOBLE, W. S. Kernel methods for predicting protein-protein interactions. *Bioinformatics*. 2005. Vol. 21, no. i, p. 38-46.

38. GIBAJA, Juan José. *Aprendizaje Estadístico con Funciones Kernel*. Universidad Nacional de Educación a Distancia, 2010.

39. SHEN, J., ZHANG, J., LUO, X., ZHU, W., YU, K., CHEN, K., LI, Y. y JIANG, H. Predicting protein-protein interactions based only on sequences information. *Proc Natl Acad Sci USA*. 2007. Vol. 104, no. 11, p. 4337-41.

40. MARTÍNEZ, José F., GARCÍA, Milton y CARRASCO, Jesús A. A survey of emerging patterns for supervised classification. *Springer Science + Business Media Dordrecht*. 2012. DOI 10.1007/s10462-012-9355-x.

41. ZAKI, N., LAZAROVA-MOLNAR, S., EL-HAJJ, W. y CAMPBELL, P. Protein-protein interaction based on pairwise similarity. *BMC Bioinformatics*. 2009. Vol. 11, no. 150.

42. HSU, Chih-wei, CHANG, Chih-chung y LIN, Chih-jen. A Practical Guide to Support Vector Classification. . 2010. Vol. 1, no. 1, p. 1-16.

43. JR, Bock y DA., Gough. Predicting protein protein interactions from primary structure. *Bioinformatics*. 2001. Vol. 17, p. 455-60.

44. GUANGREN SHI. *Data mining and knowledge discovery for geoscientist*. Elsevier Inc, 2014. ISBN 9780124104372.

45. CAI, C.Z., HAN, L.Y., JI, Z.L., CHEN, X. y CHEN, Y.Z. SVM-Prot: web-based support vector machine software for functional classification of a protein from its primary sequence. *Nucleic Acids Research*. 2003. Vol. 31, p. 3692-3697.

46. WITTEK, Peter. *Quantum Machine Learning. What Quantum Computing Means to Data Mining*. 2014. ISBN 9780128009536.

47. KANDASWAMY, K.K., PUGALENTHI, G., SUGANTHAN, P.N. y GANGAL, R. SVMCRYS: an SVM approach for the prediction of protein crystallization propensity from protein sequence. *Protein Pept. Lett*. 2010. Vol. 17, p. 423–430.

48. SUGIYAMA, Masashi. Support vector classification. In : *Introduction to Statistical Machine Learning*. 2016. p. 303-320. ISBN 9780128021217.

49. BHASKAR, Harish, HOYLE, David C. y SINGH, Sameer. Machine learning in bioinformatics : A brief survey and recommendations for practitioners. *Computers in Biology and Medicine*. 2006. Vol. 36. DOI 10.1016/j.compbiomed.2005.09.002.

50. PEREIRA, Americo y OTTO, Jan. *Review of feature selection techniques in bioinformatics by Yvan Saeys , Iñaki Inza and Pedro Larrañaga*. [no date].

51. GUYON, Isabelle y ELISSEEFF, André. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*. 2003. Vol. 3, p. 1157-1182.

52. KENT, John T. Information gain and a general measure of correlation. *Biometrika*. 1983. Vol. 70, no. 1, p. 163-173.

53. MITCHELL, Tom M. *Machine Learning*. McGraw-Hill Science. Engineering. Math, 1997. ISBN 0070428077.

54. SEIFODDINI, Hamid K. Single linkage versus Average linkage clustering in machine cells formation applications. *University of Wisconsin. Milwaukee*. 1989. Vol. 16, no. 3, p. 419-426.

55. RESTREPO, Luis F y GONZÁLEZ, Julián. De Pearson a Spearman. *Revista Colombiana de Ciencias Pecuarias*. 2007. Vol. 20, no. 2.

56. KOHAVI, Ron y JOHN, H. Artificial Intelligence Wrappers for feature subset selection. *Artificial Intelligence*. 1997. Vol. 97, no. 97, p. 273-324.

57. SAEYS, Yvan, INZA, Iñaki y LARRAÑAGA, Pedro. A review of feature selection techniques in bioinformatics. *Bioinformatics*. Vol. 23, no. 19, p. 2507-2517.

58. KITTLER, Josef. Pattern recognition and signal processing. In : . Netherlands, 1978. p. 41-60.

59. ELKAN, Charles. Evaluating Classifiers. *University of California, San Diego*. 2012.

60. HÖPPNER, Frank. *Fuzzy cluster analysis: methods for classification, data analysis and image recognition*. 1999. ISBN 04719886642.

61. VALENTE, Jose y PEDRYCZ, Witold (eds.). *Advances in Fuzzy Clustering and its Applications*. 2007. ISBN 978-0-470-02760-8.

62. GARRE, Miguel, CUADRADO, Juán José y SICILIA, Miguel Ángel. Comparación de diferentes algoritmos de clustering en la estimación de coste en el desarrollo de

software. . 2005.

63. BRADLEY, P S, FAYYAD, Usama y REINA, Cory. Scaling Clustering Algorithms to Large Databases. *Microsoft Research*. 1998. P. 1-7.

64. MCQUEEN, J. Some methods for classification and analysis of multivariate observations. In : *In 5th Berkeley Symposium on Mathematics*. 1967.

65. BERRY, M. W. Survey of Text mining: Clustering, Classification, and Retrieval. *Springer Verlag, New York, NY, USA.* 2004.

66. HOCHBAUM, D. S y SHMOYS, D. A best possible heuristic for the k-center problem. *Arizona State University*. 1985. Vol. 10, no. 2, p. 180-184.

67. LIU, Y., CAI, J., YIN, J. y HUANG, Z. An efficient clustering algorithm for small text documents. In : *In Seventh International Conference on Web-Age Information Management (WAIM 2006). IEEE Commnunications Society.* 2006.

68. REFAEILZADEH, Payam, TANG, Lei y LIU, Huan. *Cross-Validation*.

69. HONG, Robert V., MCKEAN, Joeseph y CRAIG, Allen T. *Introduction to Mathematical Stadistics*. Pearson Education, 2005.

70. SHAO, Zhifei y ER, Meng Joo. Efficient Leave-One-Out Cross-Validation-Based Regularized Extreme Learning Machine. *Neurocomputing* [online]. 2016. DOI 10.1016/j.neucom.2016.02.058. Available from: http://dx.doi.org/10.1016/j.neucom.2016.02.058

71. SATCHELL, Stephen y XIA, Wei. Analytic models of the ROC Curve Applications to credit rating model validation. In : *The Analytics of Risk Model Validation*. 2008. p. 113–133.

72. LI, Weizhong y GODZIK, Adam. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*. 2006. Vol. 22:1658-9.

73. LASSMANN, Timo, FRINGS, Oliver y SONNHAMMER, E. L L. Kalign2: High-performance multiple alignment of protein and nucleotide sequences allowing external features. *Nucleic Acids Research*. 2009. Vol. 37, no. 3, p. 858-865. DOI 10.1093/nar/gkn1006.

74. MOUSTAFA, Nourelhuda, ELHOSSEINI, Moustafa, HOSNY, Tarek y SALEM, Mofreh. Fragmented protein sequence alignment using two-layer particle swarm optimization (FTLPSO). *Journal of King Saud University - Science*. 2016. DOI 10.1016/j.jksus.2016.04.007.

75. MOUNT, David W. *Bioinformatics. Sequences and Genomes Analysis*. Cold Spring Harbor Laboratory Press, [no date].

76. WITTEN, Ian H., FRANK, Eibe y HALL, Mark A. *Data Mining. Practical Machine Learning Tools and Techniques*. United States, 2011. ISBN 9780123748560.

77. WITTEN, Ian H., FRANK, Eibe, TRIGG, Leonard E., HALL, Mark A., HOLMES, Geoffrey y CUNNINGHAM, Sally Jo. Weka: Practical machine learning tools and techniques with Java implementations. *Hamilton, New Zealand: University of Waikato, Department of Computer Science*. 1999.

78. HALL, Mark, FRANK, Eibe, HOLMES, Geoffrey, PFAHRINGER, Bernhard, REUTEMANN, Peter y IAN H. WITTEN. The WEKA data mining software: an update. . 2009. Vol. 11, no. 1, p. 10-18.

79. GOLDBERG, David E. *Genetic Algorithms in Search, Optimization and Machine Learning*.

1989.

80. PEARL, Judea. Intelligent Search Strategies for Computer Problem Solving. *Addison-Wesley*. 1984. P. 48.

81. KORF, Richard E. *Artificial intelligence search algorithms*. Atallah, Mikhail J, 1999. ISBN 0849326494.

82. MOLLINEDA, R A. Theoretical Analysis of a Performance Measure for Imbalanced Data. In : *International Conference on Pattern Recognition*. 2010. p. 621-624.

83. PRESSMAN, Roger S. *Ingeniería del software: un enfoque práctico*. 1993.

84. MORALES, Alejandro. *Construcción de sistemas multiclasificadores usando Algoritmos Genéticos y medidas de diversidad*. Universidad Central «Marta Abreu» de Las Villas, 2014.

85. KUNCHEVA, Ludmila I. y WHITAKER, Christopher J. Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy. . 2003. Vol. 51, no. 2, p. 181-207. DOI DOI: 10.1023/A:1022859003006.

86. CUNNINGHAM, Pádraig y CARNEY, John. Diversity versus Quality in Classification Ensembles Based on Feature Selection. *Machine Learning. ECML*. 2000. P. 109-116.

87. KOHAVI, Ron y WOLPERT, David H. Bias Plus Variance Decomposition for Zero-One Loss Functions 3 Bias Plus Variance for Zero-One. . 1996.

88. KUNCHEVA, Ludmila I. *Combining Pattern Classifiers Methods and Algorithms*. 2004. ISBN 9786468600.

Anexos

Anexo 1. Selección de atributos con el kernel Polinómico.

Gs(U)_ES_PRT_TI50
Pa_ES_NPR_TI50
ISA_ES_NPR_TI50
ECI_ES_PRT_TI50
Z3_ES_BSR_TI50
Gw(U)_ES_PRT_TI50
IP_ES_GLY_TI50
Mw_ES_ILE_TI50
W(U)_ES_ALA_TI50
Pt_ES_ARG_SI50
W(U)_ES_PHE_SI50
Pa_ES_PHE_SI50
Pt_ES_PCR_SI50
IP_ES_GLY_SI50
Ap_ES_PCR_TI50
IP_ES_AHR_TI50
W(U)_ES_CYS_SI50
Mw_ES_ALR_MI50
Pb_ES_MET_SI50

Anexo 2. Atributos seleccionados con Best First y el kernel RBF

Gs(U)_ES_PRT_TI50
Pa_ES_NPR_TI50
ISA_ES_NPR_TI50
ECI_ES_PRT_TI50
Z3_ES_BSR_TI50
Gw(U)_ES_PRT_TI50
IP_ES_GLY_TI50
Mw_ES_ILE_TI50
W(U)_ES_ALA_TI50
W(U)_ES_PHE_SI50
Pa_ES_PHE_SI50
IP_ES_GLY_SI50
IP_ES_AHR_TI50
W(U)_ES_CYS_SI50
Mw_ES_ALR_MI50
Pb_ES_MET_SI50

Anexo 3. Experimento sobre datas de secuencias aleatoria.

Primeramente se realizó una modelación inicial con la data obtenida de la forma explicada con los atributos derivados de la selección aplicada con el kernel Polikernel y el método “BestFirst” en la sección 3.2. La configuración de parámetros utilizada fue la que aparece por defecto para esta función: C=1 y E=1. Como resultado la Exactitud obtenida en Validación Cruzada de 10 pliegues y con todo el conjunto de datos fue de 69.48 % y 69.62% respectivamente.

Posteriormente 10 datas diferentes fueron creadas con secuencias seleccionadas de forma aleatoria. Luego se calcularon nuevamente los descriptores para cada par de dominio, solo de aquellos atributos obtenidos de la selección realizada con el Polikernel y el método de búsqueda “BestFirst”. A continuación se modeló cada una de las datas con los valores de C=1 y E=1. Los resultados de la exactitud de cada modelo se muestran en la Tabla 6.

Tabla 6. Exactitud obtenida en 10 datas de secuencias seleccionadas aleatoriamente.

Data	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
TS	68.09	68.54	68.09	67.88	68.33	68.42	68.48	68.69	68.15	67.79
CV	67.82	68.33	67.75	67.82	68.27	68.48	67.46	68.54	68.18	67.82

A primera vista se observa que la diferencia entre la exactitud de los modelos obtenidos con secuencias seleccionadas aleatoriamente, con respecto a la exactitud que se obtiene con la data antes creada es poco significativa. No obstante existen un conjunto de métricas para la medir la diversidad de los clasificadores que son útiles para estimar en qué porciento difieren. En la tesis presentada por Alejandro Morales Hernández (84) se citan un conjunto de medidas de diversidad que se basan en todo el conjunto considerando a todos los clasificadores a la vez y calculan un único valor de diversidad para todo el conjunto. Para este trabajo se seleccionaron tres métricas, donde no se le atribuye importancia a la clasificación correcta de los casos, sino que para cada caso la clasificación sea en su mayoría la misma.

Entropía

La medida de Entropía (The Entropy Measure) (85) se basa en la idea intuitiva de que en un conjunto de N casos y L clasificadores la mayor diversidad se obtendrá si $L/2$ de los clasificadores clasifican una instancia correctamente y los otros $L - L/2$ la clasifican incorrectamente. (86)

$$E = \frac{1}{N}\frac{2}{L-1}\sum_{j=1}^{N} min\left\{\left(\sum_{i=1}^{L} y_{j,i}\right), \left(L - \sum_{i=1}^{L} y_{j,i}\right)\right\}, y_{j,i} \in \{0,1\}, 0 \leq E \leq 1$$

Donde $y_{j,i}$ tendrá valor 1 si el clasificador i clasificó correctamente el caso j y 0 en caso contrario. Si E tiene valor 0 esto indica que no hay diferencia entre los clasificadores y un valor 1 indica la mayor diversidad posible.

Varianza de Kohavi-Wolpert

La varianza de Kohavi-Wolpert (Kohavi-Wolpert Variance), fue inicialmente propuesta por Kohavi y Wolpert (87). Esta medida es originada de la descomposición de la varianza del sesgo del error de un clasificador.

Kuncheva y Whitaker presentaron en (85) una modificación para medir la diversidad de un ensamblado compuesto por clasificadores binarios, quedando la medida de diversidad como:

$$KW = \frac{1}{NL^2}\sum_{j=1}^{N} Y(Z_j)\big(L - Y(Z_j)\big), 0 \leq KW \leq 1 \; donde \; Y\big(Z_j\big) = \sum_{i=1}^{L} y_{i,j}$$

Con esta medida, la diversidad disminuye a medida que el valor de KW decrece.

Medida de variabilidad

Esta medida (The Measure of Variability) tiene en cuenta si las clases asignadas por los clasificadores en cada instancia son distintas o no. Mientras mayor sea su valor, mayor será la diversidad.

$$Var = \frac{\sum_{i=1}^{N} a}{N} \; donde \; a = \begin{cases} 0 \; si \; E_1(i) = E_2(i) = \cdots = E_L(i) \\ 1 \; e.o.c \end{cases}$$

Donde N es el total de instancias y $E_L(i)$ es la etiqueta (clase) asignada a la instancia i, por el clasificador $l - ésimo$ (88)

Esta métrica parece buena para comparar pocos clasificadores. Sin embargo cuando el número de clasificadores aumenta puede mostrar cierta limitación. Por ejemplo, para el caso en que 9 de los 10 clasificadores predigan igual, y el

restante distinto, el parámetro ‘a’ valdría 1 y aportaría a la variabilidad total el mismo valor que si 5 clasificadores predijeran una clase y los otros 5 la otra.

Es por ello que para este trabajo se propone realizar una generalización para comparar clasificadores binarios donde el parámetro ‘a’ sea calculado de la siguiente forma:

$$a = \frac{L - |L_0 - L_1|}{L}$$

Donde L es la cantidad de clasificadores, L_0 y L_1 son las cantidades de clasificadores que asignan las clases ‘0’ y ‘1’ respectivamente.

La aplicación de las tres métricas (Ver Tabla 7) propuestas anteriormente evidenció la existencia de poca diversidad entre los clasificadores obtenidos con los 10 conjuntos de datos generados aleatoriamente.

Tabla 7. Resultados del estudio de diversidad de los clasificadores.

Medida de diversidad	***Valor***
Entropía	0,027
Kohavi Wolperd	0,009
Medida de variabilidad	0,054
Medida de variabilidad 2	0,025

La métrica *Medida de variabilidad 2* en la Tabla 7 surge de la aplicación de la modificación propuesta a la métrica *Medida de variabilidad*; su valor muestra como al realizar el cambio planteado la diversidad disminuye aún más. Estos resultados reflejan la poca variabilidad de los descriptores al representar secuencias de un mismo dominio, a la vez que demuestran la factibilidad de seleccionar una secuencia representativa del espacio de secuencias de cada dominio para crear el conjunto de datos a aprender por el clasificador.

Anexo 4. Diagrama de clases. Reentrenamiento del modelo.

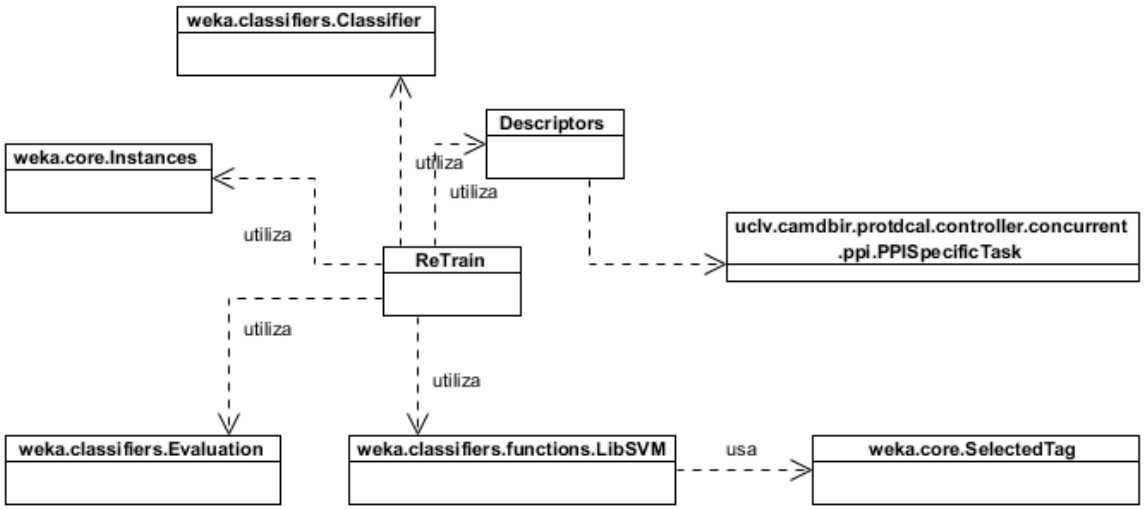

Figura 17. Diagrama de clases. Reentrenamiento del modelo.