# Universidad de las Ciencias Informáticas

# Facultad 6


**Título:** Predicción de actividad anticancerígena de compuestos orgánicos partiendo de descriptores, utilizando Máquinas de Soporte Vectorial.

**Trabajo de Diploma para optar por el título de**
**Ingeniero en Ciencias Informáticas**

**Autores:**

Yindra Thomas Abreu.

Mónica Teresa Llorente Quesada.

**Tutores:**

Dr. Ramón Carrasco Velar.

MSc. Aurelio Antelo Collado.

MSc. Olga C. Fontova de los Reyes

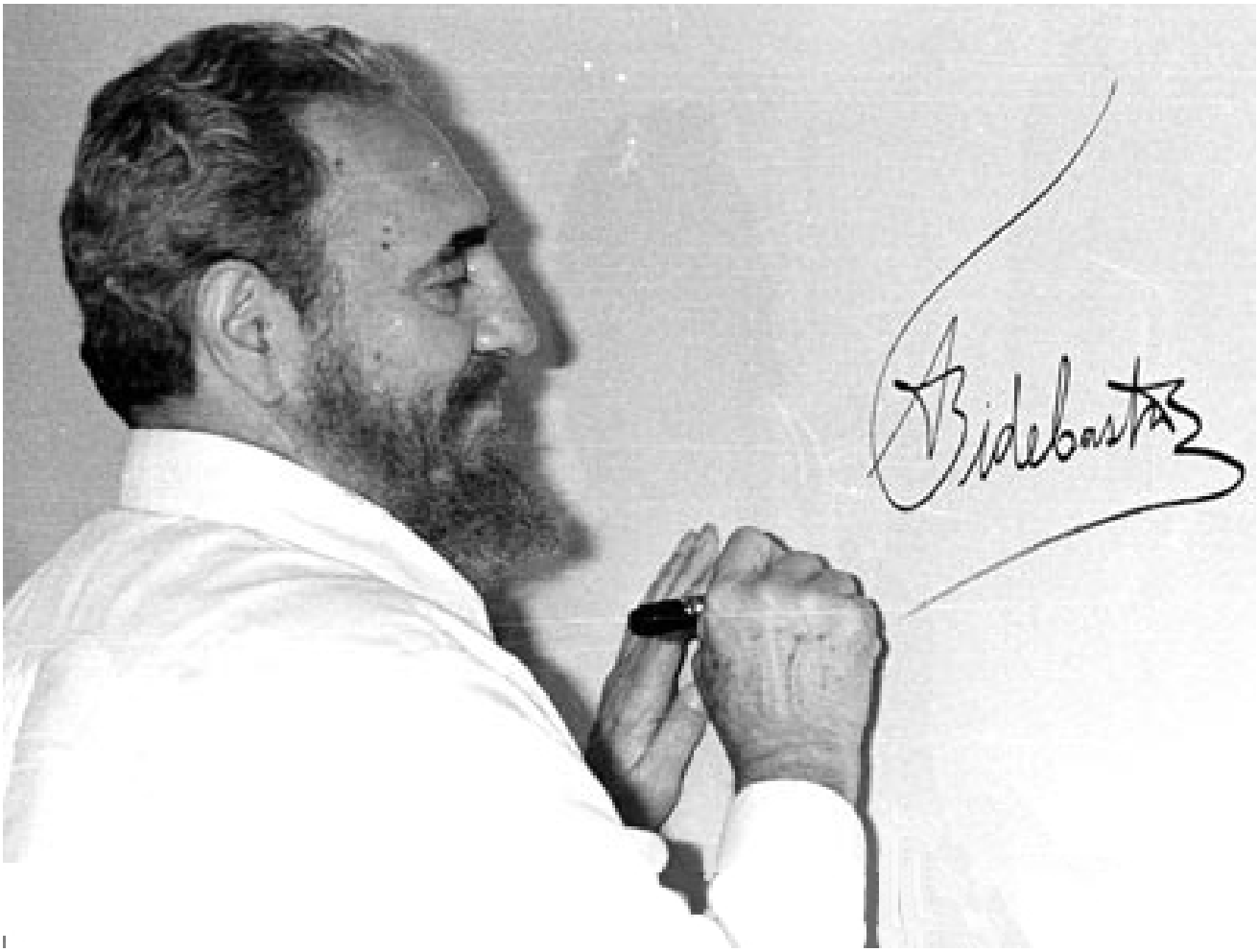
**Consultores:**

Dra. Isneri Talavera Bustamante

MSc. Yanet Villanueva Armenteros

Ing. Noslen Hernández González

**Ciudad de La Habana, Julio 2007**




***"Revolución es sentido del momento histórico; es cambiar todo lo que debe ser cambiado; es igualdad y libertad plenas; es ser tratado y tratar a los demás como seres humanos; es emanciparnos por nosotros mismos y con nuestros propios esfuerzos; es desafiar poderosas fuerzas dominantes dentro y fuera del ámbito social y nacional; es defender valores en los que se cree al precio de cualquier sacrificio; es modestia, desinterés, altruismo, solidaridad y heroísmo; es luchar con audacia, inteligencia y realismo; es no mentir jamás ni violar principios éticos; es convicción profunda de que no existe fuerza en el mundo capaz de aplastar la fuerza de la verdad y las ideas, Revolución es unidad, es independencia, es luchar por nuestros sueños de justicia para Cuba y para el mundo, que es la base de nuestro patriotismo, nuestro socialismo y nuestro internacionalismo".***

**Declaración de autoría**

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los 2 días del mes de julio del año 2007.

| | |
|---|---|
| _____ | _____ |
| Mónica Teresa Llorente Quesada | Yindra Thomas Abreu |
| _____ | _____ |
| Msc. Aurelio Antelo Collado | Dr. Ramón Carrasco Velar |

## DEDICATORIAS

***De Mónica:***

*En primer lugar a mi Comandante y a mi Revolución por darme esta maravillosa oportunidad de prepararme y brindarle mi disposición por el resto de mi vida. Gracias por demostrarme que* ***...no hacen faltas alas...*** *para ir* ***...en busca de un sueño...*** *y hacerlo realidad.*

*A mi mamita, en cualquier lugar en donde esté y sabiendo que siempre está conmigo, por ella y para ella toda mi alegría en este día y mi esfuerzo y voluntad en estos 5 años.*

*A mi tía Elena por lo orgullosa que se hubiera sentido.*

*A mis abuelos y mi tía Karen, por ser mis padres en todo momento y formarme para que pudiera llegar a ser lo que hoy soy.*

*A mi hermanita Karen porque quisiera ser su mayor ejemplo.*

*A Héctor y Delly por dejarme ser su hija mayor y en todo momento, los buenos y los malos, estar presente como mis padres, pueden estar seguros que ya lo serán para toda la vida.*

*A mi Ottico bello, por devolverme la sonrisa cuando más lo necesitaba, por su paciencia sobre todo.*

*A mi amiga y compañera de Tesis, que se merece lo mejor de este mundo por ser tan maravillosa. Ojalá y la vida quiera que sigamos cerca una de la otra para seguir ayudándonos y apoyándonos en todo momento.*

*A todos mis compañeros, en especial los que me acompañaron en primer año, aunque ya no estemos juntos la mayoría y sobre todo los que me acompañan hoy. También a las niñas del Apto*

*A mis profes, porque con la ayuda de todos, uno se olvida lo lejos que esta del hogar y sin ellos todo hubiera sido más difícil.*

*A toda mi familia.*

***De Yindra:***

*La magnitud de un trabajo como este requiere del concurso y apoyo de muchas personas a quienes dedico los resultados del mismo.*

*A mis papás Jorge y Dionisio y a mi abuela, en cualquier lugar donde estén todo mi amor y mi alegría en este día.*

*A mi mamita linda por su bondad y amor infinitos; por su maravillosa presencia, por ser siempre mi mayor ejemplo, por su gracia y cuidados, por vestirme con coraza de fe y esperanza, por amarme tanto; por la vida. Te amo.*

*A mi abuelita Tata: Por su esmero, constancia y dedicación, por sus consejos siempre sabios, por todo su cariño.*

*A mi querido hermano Jorgito por ser tan especial, por quererme mucho y cuidarme.*

*A mi querido amor Papito: Por su paciencia, por cuidarme y quererme siempre y hacer que estos 5 años pasaran más lindos. Para ti todo mi amor y felicidad.*

*A mis dos estrellitas Sandra Violeta y Jorge Armando.*

*A toda mi familia por su apoyo y confianza.*

*A todos mis profesores: Por indicarme qué dirección seguir; por alimentarme cada amanecer con el pan de la enseñanza; por proporcionarme las herramientas para caminar por los senderos de la vida; por la luz compartida con tanta ternura.*

*A Alexis por ser un padre, un amigo, por hacerme llegar con ternura cada consejo. Gracias por tu cariño.*

*A todos mis amigos, mis compañeras de 1er año, mis actuales compañeras, por haber coincidido juntos en la vida; por experimentar las tormentas, las aguas calmadas, por tantos momentos que nunca olvidaré.*

*A mi amiga y compañera de tesis por su cariño, cuidado, responsabilidad y alegría. Gracias por tantos momentos lindos.*

*Al forjador de un sueño. A quien con pasos seguros y botas de siete leguas me ha inspirado en cada empeño mostrando un corazón y fortaleza de gigante. Al Comandante de la Tropa de Futuro: !Ordene!*

# AGRADECIMIENTOS

*Queremos agradecerles a todos los que hicieron posible que este camino se nos tornara menos engorroso. A nuestros tutores, Aurelio y Carrasco por la paciencia y la comprensión, sin dejar a un lado la constancia y el deseo de seguir adelante. A Noslen e Isneris, que desde el CENATAV nos brindaron todo su apoyo. A nuestra Decana Yanet, que nunca nos dejó de la mano. A nuestros compañeros de trabajo Andy, Yaikiel y Javier por ayudarnos en todo momento. A todos nuestros compañeros de forma general.*

## RESUMEN

La presente investigación surge en el marco del proyecto conjunto entre el Centro de Química Farmacéutica y la Facultad 6 de la Universidad de las Ciencias Informáticas denominado “Una Plataforma Inteligente para la Predicción de Actividades Biológicas de Compuestos Orgánicos”, en la cual se han desarrollado un conjunto de módulos que trabajan independientes. Se pueden utilizar a través de una interfaz única que además permite la edición (2D) y visualización (3D) de moléculas. El presente trabajo forma parte del Módulo de Inteligencia Artificial (II), capaz de predecir la actividad biológica anticancerígena de la molécula a partir de descriptores moleculares. Se analizó, diseñó e implementó una aplicación para la predicción de actividad biológica empleando Máquinas de Soporte Vectorial. Para desarrollar los modelos de clasificación y regresión, se evaluaron muestras de compuestos tomados de dos ensayos diferentes de la base de datos del National Cancer Institute. Ambas muestras se dividieron en muestras de entrenamiento y de prueba en relación 8x1 en cuanto al número de compuestos de cada una. El modelo de clasificación predijo con un 96.22% y 93.52% de acierto para las muestras de entrenamiento y de prueba respectivamente.



**Palabras Claves:** Máquinas de Soporte Vectorial, predecir, actividad biológica anticancerígena, descriptores.

# Índice

## INTRODUCCIÓN

La presente investigación surge en el marco del proyecto conjunto entre el Centro de Química Farmacéutica y la Facultad 6 de la Universidad de las Ciencias Informáticas denominado “Una Plataforma Inteligente para la Predicción de Actividades Biológicas de Compuestos Orgánicos”, en la cual se han desarrollado un conjunto de módulos que trabajan independientes. Se pueden utilizar a través de una interfaz única que además permite la edición (2D) y visualización (3D) de moléculas. El presente trabajo forma parte del Módulo de Inteligencia Artificial.

Los fármacos se obtenían antiguamente a partir de plantas medicinales o de fuente animal, de donde se extrajeron los principios activos responsables de sus efectos curativos. Posteriormente, no sólo se logró identificar a los compuestos responsables de la actividad farmacológica, sino que además se lograron obtener derivados químicos sintetizados que de cierto modo imitaran la acción del producto natural [1].

Obtener un fármaco, independientemente del método que se utilice, es un proceso largo, complejo, costoso y arriesgado. En la actualidad, para la obtención de fármacos se utilizan herramientas que mejoran los procedimientos de extracción y síntesis de fármacos. Se aplica la biotecnología para aumentar los rendimientos a partir de la generación de plantas micropropagadas. La química computacional ha desarrollado la simulación de los requerimientos físico-químicos de los fármacos, para lograr especificidad sobre receptores o enzimas que correspondan a los sitios de acción. La química combinatoria, a su vez, permite generar mayor número de derivados en corto tiempo, a diferencia de las síntesis realizadas en laboratorios convencionales.

Una vez que se generan los principios activos potenciales, se evalúa su efecto farmacológico en diferentes modelos biológicos, para evidenciar su eficacia, descartando aquellos que no cubren las especificaciones farmacocinéticas y farmacodinámicas. Las moléculas que no presentan dificultad en este aspecto y que logran eliminarse sin generar toxicidad pasan al proceso de la preparación farmacéutica para posteriormente ofrecerlos al mercado.

El diseño, desarrollo y obtención de fármacos, se consolida por el desarrollo paralelo e integración de la química, la biología, la medicina, la ingeniería y la biotecnología entre otras disciplinas, dentro de la que está la bioinformática, una ciencia emergente que utiliza la tecnología de la información para almacenar, organizar, analizar y recuperar información biológica.

La bioinformática es un área de investigación multidisciplinaria, la cual puede ser definida como la interfaz entre muchas ciencias y se aplica para dar solución a problemas complejos usando herramientas de la computación.

En los últimos años, la industria farmacéutica ha reorientado sus investigaciones hacia aquellos métodos que permitan la selección racional o el diseño de nuevos compuestos. La efectividad de estos métodos depende en gran medida de los descriptores moleculares seleccionados para caracterizar la estructura química. Se han utilizado diversos tipos de descriptores en el diseño de fármacos: químico-cuánticos, topológicos, físico-químicos, híbridos.

**Programas para el cálculo de descriptores.**

Hoy día es posible encontrar diferentes programas que calculan múltiples de estos índices teóricos basados en el grafo químico. Los más conocidos y potentes son:

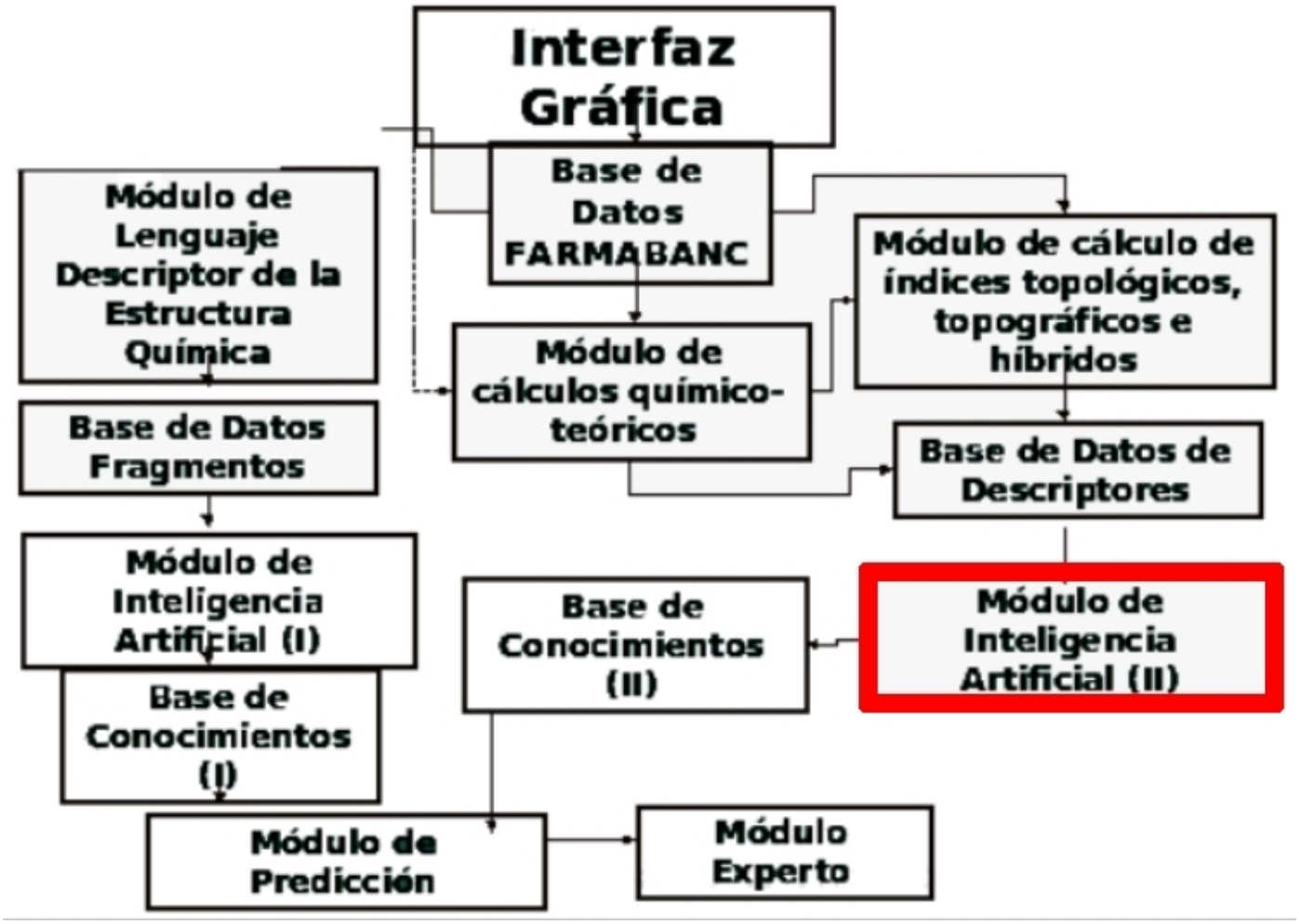
Codessa (Comprehensive Descriptors for Structural and Statistical Analysis), que brinda al usuario 116 descriptores de diferente naturaleza (8 constitucionales, 12 topológicos, 8 geométricos, 14 electrostáticos, 66 químico-teóricos y 8 termodinámicos) y un módulo para el establecimiento de modelos QSAR (Quantitative Structure Activity Relationships) basado fundamentalmente en métodos estadísticos.

Dragon, el cual calcula un total de 1 664 descriptores, de ellos 119 son descriptores topológicos, 33 índices de conectividad, 74 descriptores geométricos, entre otros. Este puede ser usado para evaluar la relación existente entre la actividad y la estructura en la molécula o las relaciones entre la propiedad y la estructura.

Molconn-Z que calcula descriptores estructurales con el fin de crear modelos QSAR para la predicción de la actividad biológica o propiedades físicas.

Todos estos programas son propietarios y de elevado costo, lo cual limita las posibilidades de adquirirlos y no se adecuan a las necesidades y características del proyecto.

**Necesidad e importancia**




Es una necesidad de la plataforma conocer cuáles son los resultados de predicción de la actividad anticancerígena utilizando diferentes técnicas de Inteligencia Artificial (IA) a partir de la descripción de la molécula por diferentes vías (fragmentos o descriptores).

Por esta razón se plantea el siguiente **problema científico**:
¿Cómo predecir la actividad biológica anticancerígena en la Plataforma?

**Objeto de estudio.**

La Inteligencia Artificial aplicada a la predicción de actividad biológica.

**Campo de acción.**

Las Máquinas de Soporte Vectorial (MSV) aplicadas al estudio de la relación estructura-actividad anticancerígena utilizando descriptores topológicos e híbridos.

**Objetivo General**

Desarrollar un módulo para la Plataforma capaz de predecir la actividad anticancerígena de compuestos orgánicos a partir de descriptores topológicos e híbridos, utilizando Máquinas de Soporte Vectorial como técnica de Inteligencia Artificial.

**Objetivos específicos:**

Analizar el módulo basado en MSV que permita predecir estructura-actividad.

Diseñar el módulo analizado.

Implementar el módulo diseñado.

Generar y validar modelos de predicción con el módulo implementado.

**Tareas generales:**

- Actualización bibliográfica acerca de las MSV.
- Definición de los principales conceptos, trabajadores, actores.
- Confección del modelo de dominio.
- Estudio de las clases de la librería libsvm para una mejor comprensión de sus funcionalidades.
- Selección de parámetros para la creación de la MSV.
- Transformación del formato de tablas de la Base de Datos al fichero que admite la MSV.
- Realizar pruebas y comparar los resultados con datos reales.

**Estructuración de la tesis.**

La tesis está conformada por Resumen, Glosario de Términos, Introducción, cuatro capítulos principales que constituyen el cuerpo de la tesis, Conclusiones, Recomendaciones y Bibliografía. Los aspectos que abarcan cada uno de los capítulos principales son:

- Capítulo 1: Fundamentación teórica.

  Se abordará en detalle todo lo relacionado con la fundamentación teórica que sustenta la presente investigación. Se presenta un estudio del estado del arte del tema y se exponen las principales tendencias, técnicas y tecnologías usadas para la implementación de la solución de software propuesta por el presente trabajo.
- Capítulo 2: Características del Sistema.

  Se mencionan los Requisitos Funcionales y no Funcionales que debe cumplir el sistema, se muestra el Modelo de Dominio propuesto para la solución del problema y se describen los Casos de Usos.
- Capítulo 3: Análisis y diseño.

  Se realiza detalladamente el Análisis y Diseño del Módulo, incluyendo los diagramas correspondientes.
- Capítulo 4: Implementación y Prueba.

  Se expone cómo se implementó el módulo y de qué forma trabaja, así como el diagrama de componentes. Se explica la generación y validación de los modelos.

# CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

La bioinformática es una ciencia frontera en la que intervienen la informática, la biología, la bioquímica, la química, la física, etc. que tiene como objetivo el manejo de grandes volúmenes de información científico-técnica, relacionada con el área de las disciplinas que la integran. Incluye la predicción de la estructura tridimensional de las proteínas a partir de su secuencia, la predicción de las funciones biológicas y biofísicas a partir de la secuencia o la estructura, la interacción entre pequeñas y grandes moléculas, el efecto de variaciones estructurales en la respuesta biológica, así como simular el metabolismo y otros procesos biológicos basados en esas funciones.

Muchos de los métodos de la computación y de las ciencias de la información sirven para estos fines, incluyendo el aprendizaje de las máquinas, la teoría de la información, la estadística, la teoría de grafos, la inteligencia artificial, los métodos estocásticos, la simulación, la lógica, etc.

## 1.1 Modelación molecular y Diseño de fármacos

Los estudios de correlación estructura química-actividad biológica por métodos computarizados permiten identificar aquellos rasgos y propiedades estructurales que son responsables de la actividad biológica de un compuesto químico, y diseñar nuevas moléculas con interés biológico para obtener en un plazo de tiempo más corto y con menos gastos nuevos compuestos potencialmente activos, lo que los convierte en una herramienta de gran importancia científicos económica.

Se requiere que las moléculas orgánicas sean convenientemente descritas para poder ser utilizadas en el diseño de fármacos. Los métodos QSPR/QSAR (Quantitative Structure-Property Relationships/ Quantitative Structure-Activity Relationships), han demostrado que las relaciones entre la estructura molecular y las propiedades físico-químicas o biológicas de los compuestos se pueden cuantificar matemáticamente con parámetros estructurales simples.

A partir de la descripción estructural de la molécula se implementan los métodos matemáticos necesarios para obtener una buena predicción de la actividad biológica. En la actualidad, se utilizan descriptores de naturaleza topológica, topográfica, químico-cuántica y químico-física para el diseño de moléculas. Recientemente, han surgido índices denominados híbridos que contienen información tanto topológica como química-física. Muchos de estos índices se basan en la descripción de la molécula como un grafo conexo en que los vértices son los átomos diferentes de hidrógeno y las aristas son los enlaces que los unen.

La representación de dichos grafos moleculares como matrices de conectividad o de distancia entre vértices o aristas del grafo, sirve como punto de partida para la generalidad de estos índices. Estos descriptores constituyen una herramienta potente para el diseño dada su facilidad de obtención y capacidad descriptiva de las moléculas, o partes de ellas.

Recientemente se desarrollaron índices híbridos, atómicos y moleculares basados en el Índice del Estado Electrotopológico de Kier y Hall y en el índice de conectividad de Randic respectivamente, los cuales incorporan información químico-física adicional a esos índices topológicos. Esto permite valorar las moléculas o partes de ellas, con un contenido estructural diferente, pues se basan, no en el grafo desprovisto de hidrógeno sino en el grafo completo. Las características anteriormente mencionadas los convierten en una herramienta simple en su concepción y facilidad de implementación con un mayor contenido de información estructural aplicable a los estudios de relación entre la estructura química y la actividad biológica.

## 1.2 ¿Qué técnicas se utilizan para esto?

Realizar el trabajo experimental, encontrar las regularidades de los datos obtenidos, y realizar la predicción para algunos fenómenos desconocidos, son los principales objetivos del trabajo de investigación en los campos de la química y de las disciplinas relacionadas.

Existe una gran tendencia a establecer un nuevo campo de la computación que integra diferentes métodos de resolución de problemas que no pueden ser descritos fácilmente mediante un enfoque algorítmico tradicional. Estos métodos tienen su origen en la emulación de sistemas biológicos. Se trata de una nueva forma de computación que es capaz de manejar las imprecisiones e incertidumbres de problemas relacionados con el mundo real. Para ello se dispone de un conjunto de técnicas de Inteligencia Artificial como la Lógica Borrosa, el Razonamiento Aproximado, Algoritmos genéticos, las Redes Neuronales, Máquinas de Soporte Vectorial, etc. Estas técnicas se pueden utilizar de forma independiente o combinada, con el objetivo de mejorar los resultados. Con estas técnicas se han realizado numerosas investigaciones en este campo, como por ejemplo:

- Se trabaja en la aplicación de diferentes modelos de Redes Neuronales Artificiales (RNA), a la solución de problemas de la Química Orgánica, especialmente en el campo de la Química Médica. Esta nueva técnica se ha integrado dentro de las ya clásicas del análisis QSAR y QSPR.
- Se han aplicado también las Máquinas de Soporte Vectorial (MSV). Estas, han demostrado

experimentalmente, que poseen capacidad predictiva comparable o superior a los modelos de MLR (regresión lineal múltiple) y RBFNN (redes neuronales con función de base radial).

Con respecto a otros algoritmos usados en la química computacional, las MSV tienen algunas ventajas excepcionales: son utilizadas tanto para la clasificación (MSVC) como para la regresión (MSVR); son aplicables para procesar datos tanto lineales como no lineales, y poseen especial capacidad de generalización.

### 1.3 ¿Qué son las Máquinas de Soporte Vectorial (MSV)?

Las MSV son en la actualidad los algoritmos que más se utilizan para todo tipo de tareas de inducción, debido a su sólido fundamento teórico y sus muchas y manejables implementaciones para el aprendizaje automático.

Las MSV son sistemas de aprendizaje que usan un espacio de hipótesis de funciones lineales en un espacio de rasgos de mayor dimensión. Esa dimensionalidad viene dada por el empleo de las funciones kernels entrenadas por un algoritmo proveniente de la teoría de optimización que se basa en un método derivado de la Teoría de Aprendizaje Estadístico.

Esta teoría fue desarrollada por Vapnik y Chervonenkis sobre la base de un problema de clasificación lineal binario e implementa el principio inductivo de Minimización del Riesgo Estructural, para obtener buenas cualidades de generalización sobre un número limitado de patrones de aprendizaje.

Dos conceptos teóricos básicos de las MSV son: la Minimización del Riesgo Empírico (MRE) que en nuestro caso lo constituye el error en la predicción de la actividad, y la Dimensión de Vapnik-Chervonenkis. La MRE es un término formal para un concepto simple: encontrar la función f(x) que minimice el riesgo promedio en el conjunto de entrenamiento.

Minimizar el riesgo empírico cuando se tiene suficientes vectores de entrenamiento no es algo difícil de hacer ya que cuando el número es muy grande se asegura que el riesgo empírico converge asintóticamente al riesgo esperado para $n \rightarrow \infty$. Sin embargo, para una muestra muy pequeña no se puede garantizar que la minimización del riesgo empírico minimice también el esperado.

La dimensión de Vapnik-Chervonenkis es una medida de la capacidad de un tipo de funciones f(α) para medir el mayor número de muestras explicables por esa familia de funciones.

**Ventajas de la MSV:**

1. Su estructura está determinada sobre la base del conjunto de entrenamiento y se necesitan pocos parámetros para entrenar.
2. El entrenamiento se reduce a la solución de un problema de optimización de una función, que se reduce a un problema de programación cuadrática convexa, por lo que la solución es completamente reproducible.
3. Es un método robusto cuando se trabaja con datos de gran dimensionalidad y los cálculos son eficientes gracias al uso de los kernels.

**Desventajas de las MSV:**

1. No es un método de modelación de clases, es decir que un "Outlier" u otro objeto va a ser clasificado obligatoriamente como perteneciente a una clase aún sin serlo.

**Problemas abiertos:**

1. Determinación o selección del kernel óptimo.

### 1.3.1 Problemas de la Química Computacional que las MSV solucionan.

Aunque los métodos clásicos de estadística se aplican con éxito en muchos campos de la química computacional, todavía existen problemas sin resolver. El origen principal de estas dificultades es que la mayor parte del conjunto de datos en problemas químicos es muy complejo. Es difícil extraer totalmente la información útil y eficiente del conjunto de datos usando métodos estadísticos clásicos, debido a las siguientes características que presentan los datos:

1) No lineales: Muchos métodos estadísticos clásicos están especialmente diseñados para problemas de procesamiento de datos lineales. Pero la mayoría de los problemas que se solucionan con el procesamiento de datos en química, son problemas que presentan datos no lineales. Por supuesto, si un conjunto de datos muestra relaciones lineales o casi lineales, el procesamiento de datos será simplificado, y los resultados del aprendizaje de la máquina serán más confiables. Sin embargo, solamente una parte pequeña del conjunto de datos en problemas prácticos se puede considerar como conjuntos lineales o casi lineales. Es razonable utilizar el coeficiente de correlación múltiple como criterio para las linealidades de la relación del conjunto de datos. Usando estos criterios, puede demostrarse que la mayor parte del conjunto de datos en problemas químicos muestran mayor o menor grado de no linealidad.

El desarrollo de las MSV ha proporcionado una nueva forma para solucionar problemas no lineales. Y en muchos casos la adaptabilidad de las MSV es mejor que otras técnicas por su amplia capacidad de generalización. Por lo tanto las MSV se consideraran una nueva herramienta de gran alcance para el procesamiento de datos en el campo de la química computacional.
Según la teoría de aprendizaje estadístico, el aprendizaje de las máquinas es el proceso de elegir la función apropiada, dentro de un conjunto dado de funciones, que correlacionen con el conjunto de datos. El conjunto de funciones usadas se llama hipótesis. Por ejemplo, en el proceso de regresión lineal o separación lineal de muestras por clasificación, todas las funciones lineales se utilizan como funciones de hipótesis. El modelo matemático construido usando la máquina de aprendizaje está obligado a seleccionar una función del conjunto de funciones de la hipótesis.

Por ejemplo, si se utiliza el método de regresión lineal como proceso de aprendizaje, el modelo matemático encontrado será seguramente lineal. Incluso, si existieran datos no lineales, estos se tratarían como residuo o ruido de la muestra y se eliminarían en el proceso de aprendizaje de la máquina.

2) Problemas con múltiples variables: las reacciones químicas están influenciadas generalmente por muchos factores, tales como la temperatura, la presión, la concentración, la actividad de los catalizadores, la clase de disolventes, la composición física o química, la composición y naturaleza de la fase, el tamaño de partícula, la presencia de impurezas, etc. Los procesos de la producción en la industria química o metalúrgica implican generalmente traspaso térmico, transferencia total, el flujo fluído y una serie de reacciones químicas, de modo que hay siempre muchos factores que influyen en las situaciones técnicas de un proceso de producción. Existen generalmente más de 5 o 6 factores que deben considerarse para solucionar un problema práctico de optimización en la industria química, y estos 5 o 6 principales factores se deben seleccionar de varias docenas de factores por vías y procedimientos para la selección de las características.

Con los métodos clásicos es difícil tratar un modelo con demasiados factores que afecten el análisis, porque la alta dimensión del espacio inducirá a la incertidumbre de los resultados.
Pero Vapnik y colaboradores demostraron que la alta dimensionalidad se puede hacer menos dañina en las MSV, proporcionando una manera eficaz para solucionar el problema de múltiples variables que presentan los datos químicos para su procesamiento.

3) Alto ruido: Uno de los requisitos de métodos estadísticos clásicos es que el ruido en un modelo debe ser lo más bajo posible, pero este requisito no se puede satisfacer en muchos casos de la química computacional. Los procesos químicos están afectados generalmente por muchos factores y es muy difícil confirmar exactamente cuántos factores se deben considerar en la solución de un problema práctico.

Por lo tanto, la influencia de factores externos que no brinden información importante serán considerados como ruido. En los procesos de producción de la industria química, los problemas de incertidumbre son más serios.

Por ejemplo, la composición de materias primas en una refinería de petróleo. La composición del petróleo crudo, cambia a menudo durante el proceso de refinación en dependencia de la fuente de esa materia prima, y la vida media y la actividad de los catalizadores pueden cambiar. Además, muchos procesos químicos exotérmicos en plantas químicas pueden inducir fenómenos caóticos. Todos estos factores son el origen de la incertidumbre o ruido en procesos de producción.

La presencia del ruido da lugar a muchas dificultades en el procesamiento de datos en la química computacional, especialmente en los problemas con tamaño de muestra pequeño. El uso de MSV no puede solucionar todos los problemas de ruido en el procesamiento de datos, pero si es posible utilizar las MSV para mejorar el ruido de la muestra. Por ejemplo, tiene formas de eliminar los outlier. Utilizando el método de validación cruzada, se pueden eliminar los datos que presentan mayor error en la predicción, y llevar a cabo la mejora de ficheros de datos. Además, la adopción de la función E-insensible (función de pérdida en MSVR) hace más robusto los modelos donde se trabajan con datos ruidosos.

## 1.4 Principio inductivo de Minimización del Riesgo Empírico (ERM).

Para minimizar el funcional de riesgo con una función de distribución $\mathrm{F(z)}$ desconocida, se aplica el siguiente principio inductivo:

1. El funcional de riesgo $R(\alpha)$ es reemplazado por el llamado Funcional de Riesgo Empírico

$$R_{emp}(\alpha) = \frac{1}{l}\sum_{i=1}^{l} Q(z_i, \alpha) \ ,$$

construido sobre la base del conjunto de entrenamiento. Esto se hace siguiendo la idea de que para obtener una buena generalización, es suficiente elegir los parámetros de la función aproximada que aseguren el número mínimo de errores sobre el conjunto de entrenamiento.

2. Se aproxima la función $Q(z, \alpha_0)$ que minimice el riesgo por la función $Q(z, \alpha_l)$ que minimice el riesgo empírico.

Este principio es llamado Principio Inductivo de la Minimización del riesgo empírico (ERM).

Decimos que un principio inductivo define un proceso de aprendizaje si para cualquier conjunto de observaciones de datos, la máquina de aprendizaje encuentra la aproximación usando este principio inductivo.

El uso del ERM en procesos de aprendizaje clásicos, como por ejemplo máquinas de aprendizaje MLP, conduce a un proceso de interpolación sobre el conjunto de entrenamiento que conlleva el fenómeno conocido como sobreentrenamiento u “overfitting”, que habitualmente es solucionado mediante el uso de criterios de parada temprana del entrenamiento (en inglés *early stopping rules*) o mediante decaimiento de pesos (en inglés *weigth decay*).

Estas técnicas son una introducción de nueva información *a priori* durante el proceso de aprendizaje, que resulta del todo necesaria para asegurar la unicidad de la solución.

Por otra parte, en numerosos problemas extraídos de la vida cotidiana los datos de aprendizaje contienen ruido, por lo que no resultaría la mejor opción realizar un proceso de aprendizaje que conduzca en su estadío final a una interpolación de los datos empíricos facilitados.

## 1.5 La dimensión de Vapnik - Chervonenkis (dimensión de VC)

Primeramente se muestra una definición de la dimensión de VC para los conjuntos de funciones indicadoras y se generaliza esta definición para conjuntos de funciones reales.

La dimensión de VC de un conjunto de funciones indicadoras $Q(z,\alpha), \alpha \in \wedge$, es el número máximo h de vectores $z_1, \ldots, z_h$ que pueden ser separados en dos clases en todas las $2^h$ formas posibles usando funciones del conjunto. Si para cualquier $n$ existe un conjunto de $n$ vectores que pueden ser separados por el conjunto $Q(z,\alpha), \alpha \in \wedge$, entonces la dimensión de VC es igual a infinito.

En la figura 1.1, cada línea discontinua separa tres vectores en dos clases. La cantidad máxima para separar tres vectores en dos clases por una función que sea una línea recta es $2^3$. Por lo tanto la dimensión VC de un sistema de líneas rectas es 3.

Por otra parte, la figura 1.2 demuestra que ninguna línea recta puede separar cuatro vectores en dos clases.

Si para cualquier **n** existe un conjunto de **n** vectores que pueden ser separados por el conjunto

$$\{f_\lambda : \lambda \in \Lambda\}.$$

entonces la dimensión de VC es igual a infinito.

La capacidad de un sistema de funciones (espacio de hipótesis) de separar un sistema de casos está estrechamente relacionado con la inducción de un espacio de hipótesis parcial. La dimensión de VC de un espacio de hipótesis, es una medida de la complejidad o expresividad de ese espacio de hipótesis. Cuanto más grande es la dimensión de VC, más grande es la capacidad (expresividad) de aprendizaje de la máquina sin error de entrenamiento. Un espacio imparcial de la hipótesis es aquel que puede representar cualquier posible concepto definido sobre una instancia del espacio ***X*** y es por lo tanto muy expresivo. Puede separar la instancia del espacio ***X***.

Sin embargo, el aprendizaje de una máquina con un espacio imparcial de la hipótesis no puede generalizar bien. Para que el aprendizaje de una máquina converja a una función final, tiene que presentar cada instancia en el espacio *X* como ejemplo de entrenamiento.

Christopher Burges escribió en el “Tutorial de las máquinas de soporte vectorial para reconocimiento de patrones”:

*“Una máquina con mucha capacidad es como un botánico con una memoria fotográfica que, cuando está frente a un árbol nuevo, concluye que no es un árbol porque tiene un número diferente de hojas del que ha visto antes; una máquina con poca capacidad es como el hermano perezoso del botánico, que declara que si es verde, es un árbol. Ninguno de los dos puede generalizar bien.”*

La teoría de la convergencia uniforme de la probabilidad desarrollada por Vapnik y Chervonenkis da un límite superior del riesgo previsto, R, con una probabilidad de 1 - ψ como

$$R(\lambda) \leq R_{emp}(\lambda) + \sqrt{\frac{h\left(\ln\frac{2l}{h}+1\right)-\ln\frac{\Psi}{4}}{l}}, \qquad \forall\lambda \in \Lambda$$

donde *h* es la dimensión de VC del *fλ* y *l* es el número de datos. El segundo término del lado derecho de la ecuación se llama la confianza de VC.

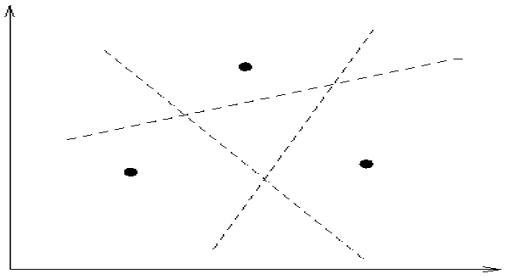


**Fig. 1.1**: Dimensión de VC

f es una línea en el plano. La dimensión de VC de {f} es 3, puesto que separan 3 vectores.

(Naturaleza de la teoría de Vapnik de aprendizaje estadístico, 1995).

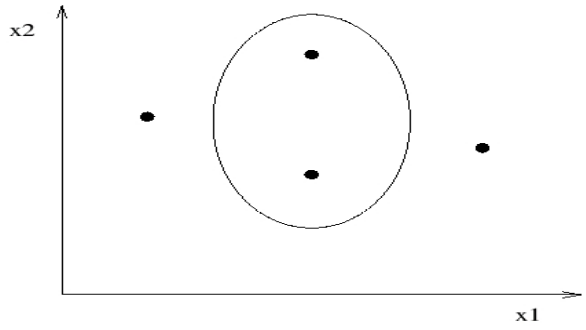



**Fig. 1.2**: Separar 4 vectores

Ninguna línea recta puede separar cuatro vectores.

(Naturaleza de la teoría de Vapnik de aprendizaje estadístico, 1995).

Para obtener un *R* pequeño, se necesita un **Remp** pequeño y un cociente pequeño de *h/l* al mismo tiempo. *h* depende del conjunto de funciones que la máquina de aprendizaje puede implementar. **Remp** depende del *fλ* elegido por la máquina de aprendizaje y disminuye con el incremento de *h*. Tenemos que elegir un *h* óptimo especialmente cuando *l*, el número de datos, es pequeño, para conseguir un buen funcionamiento. Aquí es donde aparece la minimización del riesgo estructural, que es un principio mejorado de inducción.

### 1.5.1 Dimensión de VC de un conjunto de funciones reales.

Sea $A \le Q(z,\alpha) \le B,\ \alpha \in \wedge$ , un conjunto de funciones acotadas por las constantes $A$ y $B$ ( $A$ puede ser $-\infty$ y $B$ puede ser $\infty$ )

Consideremos a lo largo del conjunto de funciones reales $Q(z,\alpha), \alpha \in \wedge$ , el conjunto de funciones indicadoras

$$I(z,\alpha,\beta) = \theta\{Q(z,\alpha) - \beta\},\ \alpha \in \wedge, \beta \in (A,B)$$

donde $\theta(z)$ es la función de paso

$$\theta(z) = \begin{cases} 0 & \text{si } z < 0 \\ 1 & \text{si } z \geq 0 \end{cases}$$

La dimensión de VC de un conjunto de funciones reales $Q(z,\alpha), \alpha \in \wedge$, se define como la dimensión de VC del conjunto de funciones indicadoras correspondientes con parámetros $\alpha \in \wedge$ y $\beta \in (A,B)$.

El funcional de riesgo estructural constituye una cota del funcional de riesgo, por lo que se trata de una condición suficiente para asegurar la minimización de $R(\alpha)$, pero en ningún caso una condición necesaria.

Para conseguir que la dimensión de VC sea una variable controlada, el principio MRE considera un esquema, semejante al principio de regularización, de espacios de aproximación anidados, del estilo $S_k = \{Q(z,\alpha), \alpha \in \wedge_k\}$ tal que $S_1 \subset S_2 \subset \ldots \subset S_n \ldots$,

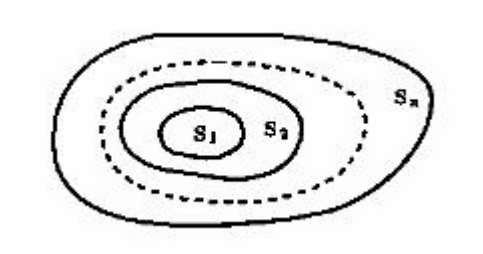



**Fig. 1.3**: Una estructura en el conjunto de funciones está determinada por subconjuntos de funciones anidadas donde los elementos de la estructura satisfacen dos propiedades:

1. La dimensión de VC $h_k$ de cada conjunto $S_k$ de funciones es finita. Por lo tanto, $h_1 \leq h_2 \ldots \leq h_n \ldots$
2. Cualquier elemento $S_k$ de la estructura contiene o un conjunto de funciones totalmente acotadas,

$$0 \leq Q(z,\alpha) \leq B_k, \alpha \in \wedge_k$$

o un conjunto de funciones que satisfacen la desigualdad

$$\sup_{\alpha \in \wedge_k} \frac{\left(\int Q^p(z,\alpha) dF(z)\right)^{\frac{1}{p}}}{\int Q(z,\alpha) dF(z)} \leq \tau_k, \qquad p > 2$$

para algún par $(p,\tau_k)$.

Para un conjunto de entrenamiento dado, el principio MRE busca la función $Q(z,\alpha_l^k)$ que minimice el riesgo empírico en el subconjunto $S_k$ para el cual el riesgo garantizado es mínimo.

El principio MRE define un compromiso entre la calidad de la aproximación a un conjunto de datos y la complejidad de la función de aproximación.

A medida que $n$ aumenta, el mínimo del riesgo empírico disminuye. Sin embargo el término responsable del intervalo de confianza aumenta. Este principio tiene en cuenta ambos factores, escogiendo el subconjunto $S_n$ para el cual la minimización del riesgo empírico garantice el mejor límite del riesgo actual.

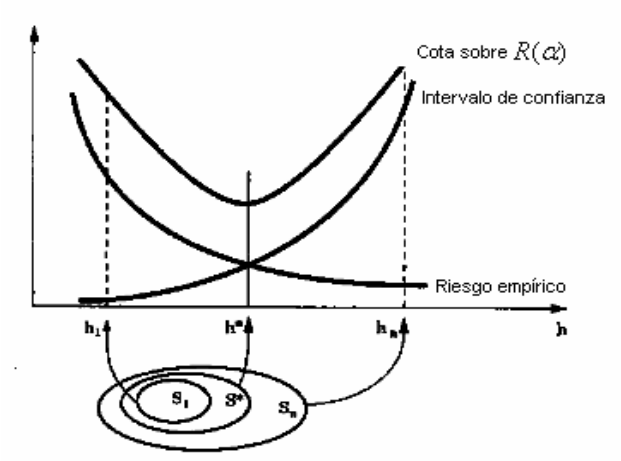



**Fig. 1.4**: El límite del riesgo de $R(\alpha)$ es la suma del riesgo empírico y del intervalo de confianza.

El riesgo empírico disminuye con el índice de los elementos de la estructura, mientras el intervalo de confianza aumenta. El menor valor para el límite del riesgo se alcanza en algún elemento apropiado de la estructura.

## 1.6 Minimización del Riesgo Estructural.

El principio inductivo de Minimización del Riesgo Estructural (MRE) es un proceso de inferencia desarrollado sobre la Teoría de Aprendizaje Estadístico, específicamente para trabajar con problemas de aprendizaje a partir de un conjunto de entrenamiento pequeño.

A partir de la obtención de una cota superior de riesgo $R(\alpha)$, válida para cualquier conjunto dado de funciones, se concluye que para asegurar su minimización, fijado el conjunto de entrenamiento, es necesario minimizar simultáneamente el funcional de riesgo empírico $R_{emp}(\alpha)$ y el factor $\phi(h_k, l)$ (el cual es directamente proporcional a la dimensión de VC e inversamente proporcional al número de elementos que componen el conjunto de entrenamiento), muchas veces llamado intervalo de confianza.

$$R(\alpha) \leq R_{emp}(\alpha) + \phi(h_k, l) = R_{srm}(\alpha)$$

Cada opción de la estructura *S* crea un algoritmo de aprendizaje. Para un conjunto dado de *l* ejemplos z1,…, zl, el principio de la minimización del riesgo estructural elige una función en el subconjunto.

$$\{f_\lambda : \lambda \in \Lambda_n\}$$

Sin embargo, implementar un MRE puede resultar difícil porque la dimensión de VC de podría ser difícil de computar. Aunque se puede computar encontrando

$$min\left(R_{emp}(\lambda) + \sqrt{\frac{h_n}{l}}\right).$$

Las MSV, pueden alcanzar la meta de reducir al mínimo el límite superior del *R(λ)* minimizando al mismo tiempo *h* y ***Remp(λ)*** en la dimensión de VC durante el entrenamiento. La estructura en la cual se basan las MSV es un sistema de separación de hiperplanos.

Sea *X* una instancia del espacio dimensional N. Tenemos un conjunto de vectores {x1, x2,...,xl} donde *xi ϵ X*. Cada hiperplano w*x-b=0 corresponde a un par canónico (par único) (w, b) con la restricción que

$$\min |w*xi-b| = 1, \qquad i=1,.....l.$$

Esto significa que el punto más cercano al hiperplano tiene una distancia de 1/||w||. La dimensión de VC de los hiperplanos canónicos es N + 1. Para que la minimización del riesgo estructural sea aplicable, se necesita construir conjuntos de hiperplanos de la dimensión de VC que varíen de modo que reduzca al mínimo la dimensión de VC y el riesgo empírico simultáneamente. Esto se alcanza agregando una restricción en **w**.

Sea *R* el radio de la esfera más pequeña *Bx1,.....xl* que contiene x1, x2,...,xl.

$$B_{\mathbf{x}_1,\ldots,\mathbf{x}_l} = \{\mathbf{x} \in X : \|\mathbf{x}-\mathbf{a}\| < R\}, \mathbf{a} \in \mathbf{X}.$$

$f_{\mathbf{w},b}$ También definimos una función de decisión de modo que

$$f_{\mathbf{w},b} : B_{\mathbf{x}_1,\ldots,\mathbf{x}_l} \longrightarrow \{\pm 1\},$$

$$f_{\mathbf{w},b} = sgn((\mathbf{w}.\mathbf{x}) - b).$$

La posibilidad de introducir una estructura en el conjunto de hiperplanos se basa en el resultado de Vapnik que el conjunto de hiperplanos canónicos

$$\left\{ f_{\mathbf{w},b} = sgn(\mathbf{w}.\mathbf{x} - b) \;\; |\|\mathbf{w}\| \leq A\} \right\}$$

tiene una dimensión de VC *h*, que satisface

$$h \leq min\{\lceil R^2 A^2 \rceil, N\} + 1.$$

Quitando la restricción ||w|| <= A se tiene un conjunto de funciones cuyo valor de la dimensión de VC es N + 1, donde N es la dimensionalidad de *X*. Agregando la restricción ||w|| <= A, se observa que el valor de las dimensiones de VC es una N mucho más pequeña y por lo tanto se trabaja en un espacio dimensional muy alto.

Geométricamente hablando, la distancia de un punto **x** al hiperplano definido (w, b) es

$$d(\mathbf{x};\mathbf{w},b) = \frac{|\mathbf{w}.\mathbf{x} - b|}{\|\mathbf{w}\|}.$$

## 1.7 Teoría de Aprendizaje Estadístico.

Los métodos estadísticos que se utilizan en química casi todos están basados en la teoría estadística clásica. Uno de los principios básicos en estadística clásica es la ley de los números grandes. Según este principio, cuando el número de observaciones tiende a infinito, la función de distribución empírica F'(x) converge a la función de distribución real F(x). Es decir para conseguir un modelo matemático confiable usando el aprendizaje de las máquinas, hay que garantizar que el conjunto de datos usados para el entrenamiento sea lo suficientemente grande como para que tienda a infinito.
Sin embargo, en cualquier problema práctico, incluyendo el aprendizaje de las máquinas para resolver tareas de química, es imposible tener tantas muestras para el entrenamiento y creación del modelo matemático. Por el contrario, en la mayor parte del trabajo de procesamiento de datos químicos el número de las muestras del entrenamiento es generalmente pequeño.
Por ejemplo, en los estudios QSAR, que es uno de los pasos más importantes para el diseño molecular, los datos que se utilizan como conjunto de entrenamiento son datos conocidos de algunos compuestos similares, y no excede de varias decenas.

Es natural hacernos la siguiente pregunta: ¿Tiene alguna influencia significativa en la confiabilidad de los modelos creados usando máquinas de aprendizaje, esta contradicción entre el principio de la ley de los números grandes y el verdadero tamaño de los datos de entrenamiento existentes?
En los últimos años, una teoría extensamente reconocida de las ciencias estadística, la Teoría de Aprendizaje Estadístico (TAE), se ha propuesto encontrar la respuesta de la pregunta antes dicha. Y varios métodos que utilizan el aprendizaje de máquinas, incluyendo la MSV y RNA, han sido propuestos basados en esta teoría. Estos nuevos métodos de cómputo se han utilizado en muchos campos, incluyendo el reconocimiento de imágenes, la clasificación de textos y en estudios de ADN, con buenos resultados.

### 1.8 ¿Cómo aprende una MSV?

El aprendizaje de una máquina es un problema de optimización. Dada la tarea de clasificar ciertos datos en un número de clases, un programa de computadora tiene que buscar una función dentro de la hipótesis, que pueda predecir correctamente la etiqueta **y** de la clase, para los datos de entrada **x** del espacio de la hipótesis.

Dado un sistema de datos $P(\mathbf{x}, y)^1$ de una distribución desconocida

$$(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_l}, y_l) \ , \mathbf{x_i} \in \mathbb{R}^n \ , y_i \in \{-1, 1\}.$$

Se tiene un sistema de funciones de decisión, o el espacio de la hipótesis

$$\{f_\lambda : \lambda \in \Lambda\}.$$

donde Λ (un índice fijado) es un sistema de parámetros abstractos, no necesariamente vectores.

$$f_\lambda : \mathbb{R}^n \longrightarrow \{-1, +1\}.$$

*fλ* también es llamado hipótesis.

Dada una función fλ, el riesgo esperado (el error de prueba) R(λ) es el error medio posible de *fλ* en el ejemplo desconocido dibujado aleatoriamente de la distribución muestra P(x, y).

$$R(\lambda) = \int \frac{1}{2} |f_\lambda(\mathbf{x}) - y| dP(\mathbf{x}, y).$$

R(λ) es la medida en que la hipótesis *fλ* predice de manera correcta la etiqueta **y** de una entrada **x**. Puesto que la distribución de la muestra P es desconocida, hay que utilizar principios inductivos para minimizar el riesgo. Esto se logra muestreando los datos y computando una aproximación estocástica al riesgo real. Esta aproximación se llama riesgo empírico Remp(λ) (error del entrenamiento)

$$R_{emp}(\lambda) = \frac{1}{l} \sum_{i=1}^{l} |f_\lambda(\mathbf{x_i}) - y_i|,$$

donde *l* es el número de muestras.

## 1.9 ¿Cómo trabajan las MSV para clasificar?

La ecuación general de un plano en n-dimensiones es w*x = b donde x es un n x 1 vector, **w** es la normal al hiperplano y **b** es una constante (escalar). De todos los puntos en el plano, existe uno que tiene una distancia mínima *dmin* al origen

$$d_{min} = \frac{|b|}{||\mathbf{w}||}.$$

Los patrones del entrenamiento (x1,y1),...,(xl,yl) están presentes donde ***xi*** es un vector de dimensión *d* y:

yi=1 si xi esta en la clase A,

yi=-1 si xi esta en la clase B.

Si los datos son linealmente separables, entonces existe un vector d-dimensional **w** y un escalar *b* tales que:

w*xi-b >= 1 si yi=1;

w*xi-b <= 1 si yi=-1

En forma compacta, la ecuación se puede escribir como:

yi (w*xi-b) >= 1 ó (yi (w*xi-b)-1) < 0.

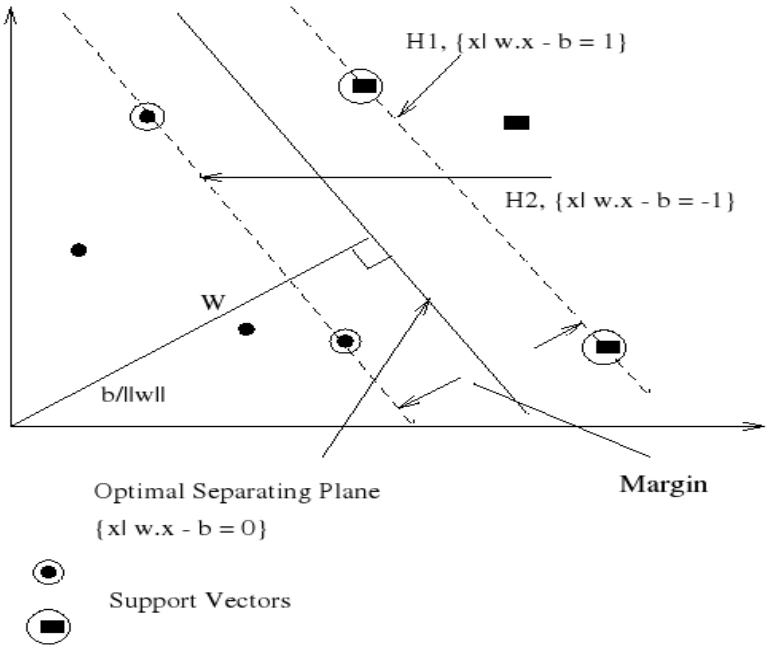



**Fig. 1.5:** Plano óptimo de separación y vectores de soporte.

(w, b) define el hiperplano que separa las dos clases de datos. La ecuación del hiperplano es w*x-b=0 donde **w** es la normal al plano, *b* es la distancia mínima del origen al plano. Para que cada superficie de desición (w, b) sea única, se normaliza la distancia perpendicular que separa el origen del hiperplano dividiéndolo por ||w||, teniendo la distancia como b/||w||.

Según lo representado (Fig. 1.5), la distancia perpendicular del origen al hiperplano H1: w*xi-b=-1 es

$$\frac{|1+b|}{||\mathbf{w}||}$$

La distancia perpendicular del origen al hiperplano H2: w*xi-b=-1 es |b-1|/||w||. Los vectores de soporte son definidos como puntos de entrenamiento en *H1* y *H2*. Si se quita algún punto en esos dos planos no cambiaría el resultado de la clasificación, pero si se quita algún vector de soporte sí. El margen, la distancia entre dos hiperplanos *H1* y *H2* es 2/||w||. El margen determina la capacidad de la máquina de aprendizaje que alternadamente determina el límite del riesgo actual - error de prueba esperado R(λ). Cuanto más ancho es el margen más pequeño es *h*.

Por lo tanto la meta es maximizar 2/||w|| que es equivalente a minimizar $\frac{\|\mathbf{w}\|^2}{2}$
Ahora se formula el problema como:

Minimizar

$$f = \frac{\|\mathbf{w}\|^2}{2}$$

donde

g = -(yi(w*xi-b)-1) <= 0, i=1,..,l

y *l* es el número de los ejemplos de entrenamiento.

Este problema puede ser solucionado usando técnicas estándares de Programación Cuadrática. Sin embargo, usando el método Lagrangiano para solucionar el problema hace más fácil ampliarlo al caso linealmente no separable.
La función Lagrangiana de este problema es

$$\begin{aligned} L_p(\mathbf{w}, b, \Lambda) &= f(\mathbf{x}) + \sum_{i=1}^{l} \lambda_i g_i(\mathbf{x}) \\ &= \frac{\mathbf{w} \cdot \mathbf{w}}{2} - \sum_{i=1}^{l} \lambda_i (y_i(\mathbf{w} \cdot \mathbf{x_i} - b) - 1) \\ &= \frac{\mathbf{w} \cdot \mathbf{w}}{2} - \sum_{i=1}^{l} \lambda_i y_i \mathbf{w} \cdot \mathbf{x_i} + \sum_{i=1}^{l} \lambda_i y_i b + \sum_{i=1}^{l} \lambda_i \end{aligned}$$

donde Λ = (λ1, λ2,....., λl) es el conjunto de los multiplicadores de Lagrange de los ejemplos de entrenamiento.

### 1.9.1 Condiciones KKT necesarias:

Condición del gradiente:

$$\frac{\partial L_p}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{l} \lambda_i y_i \mathbf{x_i} = 0,$$

$$\frac{\partial L_p}{\partial \mathbf{w}} = \left( \frac{\partial L_p}{\partial w_1}, \frac{\partial L_p}{\partial w_2}, \ldots, \frac{\partial L_p}{\partial w_d} \right)$$

$$\frac{\partial L_p}{\partial b} = \sum_{i=1}^{l} \lambda_i y_i = 0,$$

$$\frac{\partial L_p}{\partial \lambda_i} = g_i(\mathbf{x}) = 0.$$

Condición de ortogonalidad:

$$\lambda i g i = -\lambda i (y i (w^* x i - b) - 1) = 0 \qquad i = 1, \ldots, l$$

Condición de viabilidad:

$$-y i\ (w^* x i - b) + 1 <= 0 \qquad i = 1, \ldots . l$$

Condición de no-negatividad:

$$\lambda i >= 0 \qquad i = 1, \ldots \ldots, l.$$

El punto de ensilladura de Lagrange determina la solución al problema de la optimización. El problema dual es:

Maximizar

$$L_D = \sum_{i=1}^{l} \lambda_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{i=1}^{l} \lambda_i \lambda_j y_i y_j \mathbf{x_i} \cdot \mathbf{x_j}$$

donde

$$\sum_{i=1}^{l} \lambda_i y_i = 0$$

$$\lambda_i \geq 0 \qquad i = 1, \ldots, l.$$

Se pueden encontrar todos los *λ′* solucionando la ecuación con la Programación Cuadrática.
**w** se obtiene mediante la siguiente ecuación

$$\mathbf{w} = \sum_{i=1}^{l} \lambda_i y_i \mathbf{x_i}.$$

*λi* es > 0 en el punto donde un vector de soporte y su restricción correspondiente son activos. Es cero en el punto que no es un vector de soporte y su restricción correspondiente son inactivos. Podemos calcular *b* usando la ecuación

$$\lambda i\ (yi\ (w*xi-b)\ -1) = 0 \qquad i=1,.....,l$$

eligiendo el *xi* con *λ* distinto de cero.
La clase de los datos de entrada **x** se determina con: *class(x) = sign (w*x-b)*
En la práctica debido a la implementación numérica, el valor medio de *b* es encontrado calculando el promedio.

### 1.9.2 Margen y plano de separación óptimo

La cantidad $\gamma_i$ en el teorema 1.1 determina cómo se pueden separar dos clases y consecuentemente cómo converge rápidamente el algoritmo del perceptrón de aprendizaje.

**Definición 1.1:** Sea (w, b) un hiperplano determinado por una función lineal de $f : \mathbb{R}^n \to \mathbb{R}$ valores reales usada para la clasificación. Definimos el margen de un punto de la muestra (*xi, yi*) con respecto al hiperplano de cantidad

$$\gamma_i = y_i \left( \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \right).$$

Observar que

$$\gamma_i > 0$$

implica la clasificación correcta de (*xi, yi*). Según la definición 1.1, algunas de las definiciones relativas se derivan como sigue.

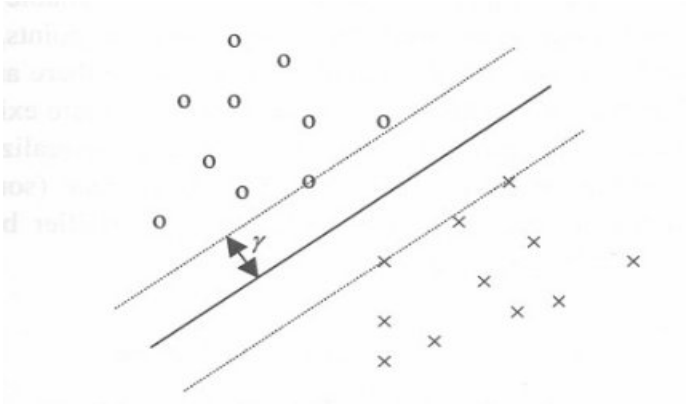



**Fig. 1.6:** El margen del conjunto del conjunto de entrenamiento en dos dimensiones.

La distribución del margen de un hiperplano (w, b) en un conjunto de entrenamiento *S* determinado, es la distribución de los márgenes de todos los puntos de muestra en *S*. Generalmente, el mínimo de la distribución del margen se refiere a cómo el margen de un hiperplano (w, b) determina un conjunto de entrenamiento S.

Margen geométrico: el margen en la definición 1.1 a veces se nombra como el funcional del margen. Para entenderlo más claramente, se introduce el margen geométrico, que es el cociente del funcional del margen al vector normalizado:

$$\gamma_{geom,i} = \frac{y_i\left(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b\right)}{\|\mathbf{w}\|}$$

El margen geométrico igualará el funcional del margen si el vector de peso es un vector unitario. El margen del conjunto de entrenamiento *S* es el margen geométrico máximo de todos los hiperplanos.

### 1.9.3 Margen máximo del hiperplano

Es bueno tener clasificadores que alcancen un margen, así la estimación es "confiable" en el entrenamiento fijado, también trabajará bien con una muestra de datos desconocidos, es decir generalizará bien. Hay muchos clasificadores lineales que pueden separar los puntos de la muestra, pero existe solamente uno que maximiza el margen geométrico. El hiperplano que realiza esto se le llama hiperplano del margen máximo (a veces conocido como hiperplano de separación óptimo), y el clasificador basado en este hiperplano se llama el clasificador del margen máximo.

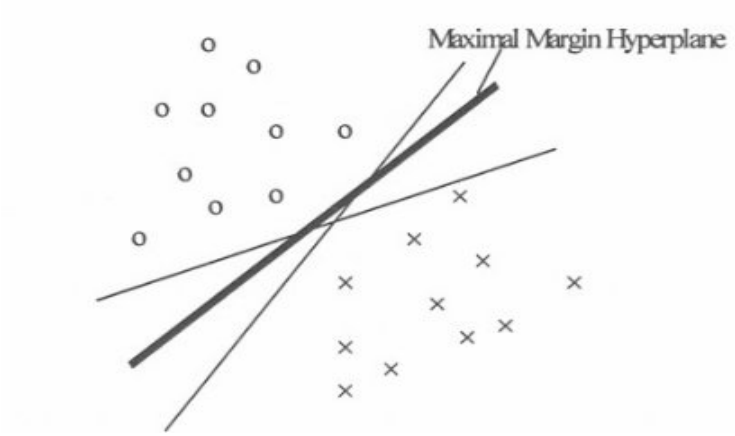



**Fig. 1.6**: Margen Máximo del Hiperplano.

Al mínimo de la distribución del margen a veces se le llama margen del hiperplano (w, b) de un conjunto entrenamiento *S*. Por lo tanto, el problema de buscar el clasificador del margen máximo está determinado si existen algunos (w*, b*) solucionando el problema siguiente:

Maximizar

$$\gamma_{\mathbf{w},b} = \min_{i=1}^{\ell} y_i f_{\mathbf{w},b}(\mathbf{x}_i) .$$

Cualquier modificación en la fórmula debe realizarse antes de los siguientes pasos. Observe que en la definición del margen funcional hay un problema. La función lineal *f (x)* asociada al hiperplano (w, b) no cambia si reescalamos el hiperplano (*λ***w**, *λb*), para

$$\lambda \in \mathbb{R}^+$$

Sin embargo, habrá un cambio en el funcional del margen en comparación con el margen geométrico. Por lo tanto, el margen geométrico es maximizado por el vector **w*** y la diagonal *b** que satisface el siguiente problema de optimización:

$$\text{maximize}_{\mathbf{w},b} \min_{i=1}^{\ell} \frac{y_i\left(\langle \mathbf{w}\cdot\mathbf{x}_i\rangle + b\right)}{\|\mathbf{w}\|}$$

$$= \min_{i=1}^{\ell} y_i \operatorname{sgn}\left(\langle \mathbf{w}\cdot\mathbf{x}_i\rangle + b\right)\left\|\frac{\langle \mathbf{w}\cdot\mathbf{x}_i\rangle + b}{\|\mathbf{w}\|}\right\|$$

$$= \min_{i=1}^{\ell} y_i \operatorname{sgn}\left(\langle \mathbf{w}\cdot\mathbf{x}_i\rangle + b\right)\left\|\frac{\langle \mathbf{w}\cdot\mathbf{x}_i\rangle + b}{\|\mathbf{w}\|}\frac{\mathbf{w}}{\|\mathbf{w}\|}\right\|$$

$$= \min_{i=1}^{\ell} y_i \operatorname{sgn}\left(\langle \mathbf{w}\cdot\mathbf{x}_i\rangle + b\right)\left\|\frac{\langle \mathbf{w}\cdot\mathbf{x}_i\rangle}{\|\mathbf{w}\|^2}\mathbf{w} + \frac{b}{\|\mathbf{w}\|^2}\mathbf{w}\right\|$$

En la derivación de las fórmulas, la salida f(x) es un escalar, por lo tanto la multiplicación por el vector unitario w/||w|| no debe cambiar la norma. La última fórmula de arriba tiene una interpretación geométrica simple, puesto que

$$\left\langle \mathbf{w}\cdot\left(-b\mathbf{w}/\|\mathbf{w}\|^2\right)\right\rangle + b = 0$$

-bw/||w|| es el vector en dirección **w** que termina a la derecha del hiperplano de decisión y para el vector de muestra *xi*, [w*xi]w/||w|| es la proyección de **xi**, sobre **w**.

Así, el problema del clasificador del margen máximo se convierte en la búsqueda apropiada de (w), la norma de las diferencias del vector

[w*xi]w/||w||-(-bw/||w||) firmadas por *yig(xi)*.

Definiendo

$$\gamma = \min_{i=1}^{\ell} y_i \left( \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \right) / \|\mathbf{w}\|$$

como el límite más bajo en el margen. El problema del máximo-mínimo se puede transformar en la tarea de optimización siguiente:

Maximizar:

$$\gamma_{\mathbf{w},b},$$

donde:

$$\frac{y_i \left( \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \right)}{\|\mathbf{w}\|} \geq \gamma$$

que es alternadamente equivalente al siguiente problema:

maximizar

$$\gamma_{\mathbf{w},b},$$

donde:

$$\|\mathbf{w}\| = 1 \quad \text{and} \quad y_i \left( \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \right) \geq \gamma$$

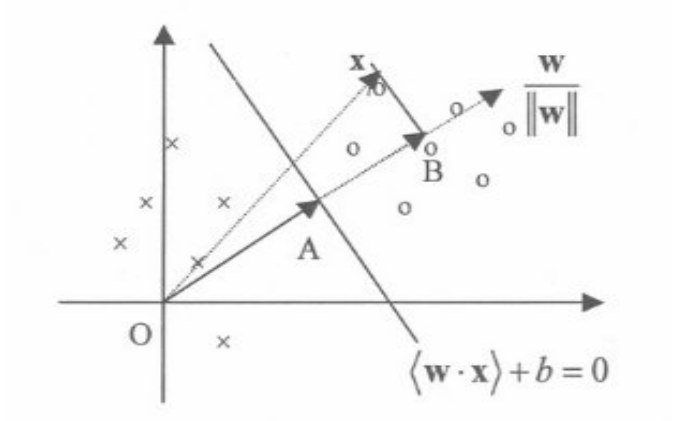



**Fig. 1.7**: Interpretación geométrica del hiperplano máximo.

Buscando un vector de peso **w** que obtenga los puntos yi[w*xi], pero con la restricción que el ancho del vector recaiga en la esfera unitaria para evitar la obtención de puntos grandes "libre" por encima de **w**. Los hiperplanos con el funcional del margen *l* se conocen como hiperplanos canónicos. Si **w** es el vector de peso que realiza un funcional del margen de *l* en el punto positivo ***x1*** y el punto negativo ***x2***, su margen geométrico se puede computar como sigue. Recordar que un funcional del margen de *l* implica

[w*x1]+b=+1,

[w*x2]+b=-1,

mientras que para computar el margen geométrico hay normalizar w. El margen geométrico $\gamma_i$ es entonces el margen funcional de clasificar el resultado

$$\gamma = \frac{1}{2}\left(\left\langle \frac{\mathbf{w}}{\|\mathbf{w}\|} \cdot \mathbf{x}_1 \right\rangle - \left\langle \frac{\mathbf{w}}{\|\mathbf{w}\|} \cdot \mathbf{x}_2 \right\rangle\right)$$

$$= \frac{1}{2\|\mathbf{w}\|}\left(\langle \mathbf{w} \cdot \mathbf{x}_1 \rangle - \langle \mathbf{w} \cdot \mathbf{x}_2 \rangle\right)$$

$$= \frac{1}{\|\mathbf{w}\|}$$

Por lo tanto, el margen geométrico será igual a 1/||w|| y podemos obtener (w*, b*) solucionando la tarea de:

Minimizar

$$||w|| \quad \text{donde} \quad yi([w*xi]+b)>=1,$$

El margen máximo es igual a la norma mínima del vector de peso y el hiperplano de separación óptimo obtenidos reduciendo al mínimo el último.

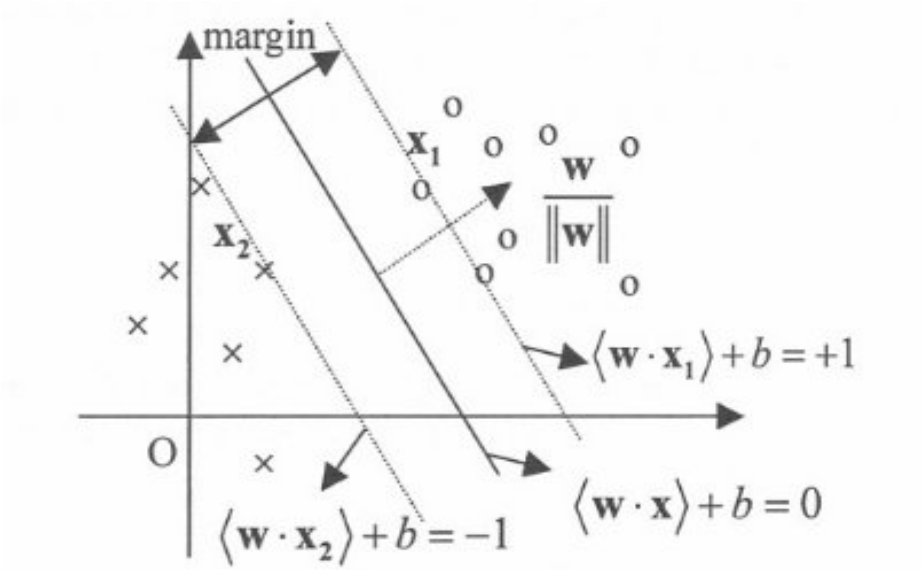


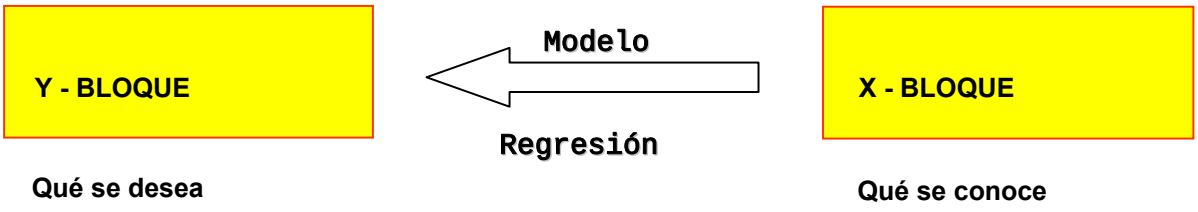

**Fig. 1.8:** Hiperplanos Canónicos

## 1.10 Regresión

Una versión de una MSV para la regresión fue propuesta en 1996 por Vapnik, Harris Drucker, Chris Burges, Linda Kaufman y Alex Smola. El modelo producido por la MSV depende solamente de un subconjunto de los datos del entrenamiento, porque la función de coste para construir el modelo no reconoce los puntos del entrenamiento que están fuera del margen especificado. Análogo, el modelo producido por una máquina de soporte para la regresión (MSR) depende solamente de un subconjunto de los datos de entrenamiento, porque la función de coste para construir el modelo obvia cualquier dato del entrenamiento que esté cercano (dentro de un ε del umbral) a la predicción modelo.

Esta técnica es aplicable al problema planteado ya que es una forma de aprendizaje supervisado y se pueden ajustar los modelos de predicción hasta encontrar el que prediga con mayor exactitud, aprovechando la característica de la máquina que a mayor cantidad de datos de entrada mejor ajuste de la ecuación.

Además, las técnicas de regresión permiten hacer predicciones sobre los valores de cierta variable ***Y*** *(dependiente)*, a partir de los valores de otra variable ***X*** *(independiente)*, entre las que existe una relación. Esto es exactamente lo que tenemos, valores conocidos de descriptores y necesitamos predecir la actividad biológica, que es desconocida hasta el momento.




**La regresión** crea un modelo para predecir rasgos numéricos desconocidos a partir de rasgos numéricos conocidos.

Es necesario decir que aunque estos procedimientos matemáticos que la fundamentan son antiguos esta técnica fue creada en el año 1996.

**Ejemplos de predicción:**

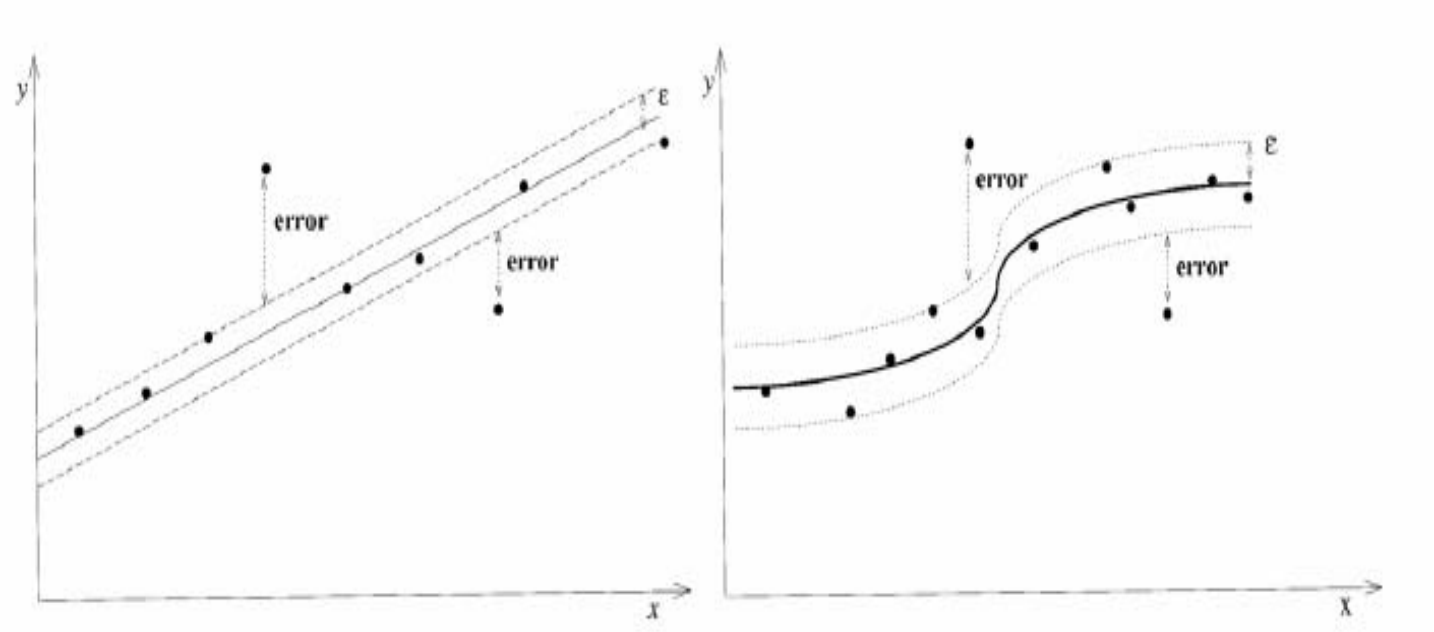
Datos conocidos o de entrenamiento para formar el modelo, donde el valor a predecir es conocido.

| moléculas \ descriptores | descriptor 1 | descriptor 2 | descriptor 3 | descriptor 4 | activ. biológ |
|---|---|---|---|---|---|
| molécula 1 | 1 | 3.6 | 2.3 | 6 | 1.5 |
| molécula 2 | 1.5 | 5 | 0.6 | 4.3 | 0.9 |
| molécula 3 | 2 | 1.6 | 3.7 | 2.8 | 1.4 |
| molécula 4 | 3.3 | 2.1 | 4 | 1.7 | 0.5 |

Datos independientes y datos a predecir (dependientes).

| moléculas \ descriptores | descriptor 1 | descriptor 2 | descriptor 3 | descriptor 4 | activ. biológ |
|---|---|---|---|---|---|
| molécula 1 | 1.7 | 0.5 | 1.6 | 3.7 | y |
| molécula 2 | 0.6 | 4.3 | 0.6 | 4.3 | y |
| molécula 3 | 3.7 | 2.8 | 3.7 | 2.8 | y |
| molécula 4 | 2.8 | 2.8 | 2 | 1.6 | y |

La regresión tiene como objetivo buscar una función que presente como máximo **ε** desviaciones de los valores de salida mientras sea tan “plana” como sea posible, es decir no preocuparán los errores en la medida que sean menores que **ε**, pero no aceptará ninguna desviación mayor que esta.




También se aplica la regresión para obtener un valor cuantitativo, un porciento de pertenencia, un número que informe que tan anticancerígena o no puede ser la actividad biológica de la molécula descrita, y no solamente una respuesta de pertenece o no pertenece, o de si es anticancerígena o no.

Las MSR se usan manteniendo todas las características principales del algoritmo de máximo margen.

- Se estima una función no lineal por la máquina de aprendizaje lineal en el espacio inducido por los kernels.
- Se controla la capacidad del sistema por un parámetro que no depende de la dimensionalidad del espacio.
- Como en el caso de clasificación, el algoritmo de aprendizaje minimiza una función convexa y su solución es esparcida.

### 1.10.1 Máquinas de Soporte para la Regresión (MSR)

El problema de la regresión es estimar (aprender) una función

$$f(\mathbf{x}, \boldsymbol{\lambda}) : \quad X(\mathbb{R}^d) \to \mathbb{R},$$

donde *X* denota el espacio de los patrones de la entrada. λ ∈ Λ, Λ es un sistema de parámetros abstractos de un sistema independiente distribuido idénticamente con muestras de tamaño N,

$$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N), \quad \mathbf{x}_i \in X(\mathbb{R}^d), \quad y_i \in \mathbb{R},$$

donde las muestras antes dichas fueron dibujadas de una distribución desconocida P (x, y).

Ahora, para encontrar la función f (x, λ∗) con el valor más pequeño posible para el riesgo previsto (o el error de la prueba) esta dada por:

$$R[\boldsymbol{\lambda}] = \int l(y, f(\mathbf{x}, \boldsymbol{\lambda}))P(\mathbf{x}, y)d\mathbf{x}dy,$$

donde *l* es la función de la pérdida.

Generalmente la probabilidad de la distribución P(x, y) es desconocida. Por lo tanto no se puede computar y reducir al mínimo, el riesgo previsto. Pero teniendo una cierta información de P(x, y) computar una aproximación estocástica de R[λ] por el riesgo empírico está dado por:

$$R_{emp}[\boldsymbol{\lambda}] = \frac{1}{N}\sum_{i=1}^{N} l(y_i, f(\mathbf{x}_i, \boldsymbol{\lambda})).$$

Esto ocurre porque la ley de números grandes garantiza que el riesgo empírico converge en probabilidad al riesgo previsto. Solamente la reducción al mínimo del riesgo empírico puede causar problemas, tales como mala valoración u overfitting, y no poder obtener buen resultado cuando entren nuevos datos.

Para solucionar el problema de la muestra pequeña, la teoría estadística o de VC, proporciona los límites entre la desviación del riesgo empírico y del riesgo previsto. Un límite uniforme típico de Vapnik y de Chervonenkis, que sostiene con probabilidad 1 − η, tiene la forma siguiente:

$$R[\boldsymbol{\lambda}] \leq R_{emp}[\boldsymbol{\lambda}] + \sqrt{\frac{h(\ln\frac{2N}{h}+1) - \ln\frac{\eta}{4}}{N}}, \quad \forall \boldsymbol{\lambda} \in \Lambda,$$

donde *h* es la dimensión de VC de f (x, λ).

De este límite, está claro que para alcanzar el riesgo pequeño previsto, es decir, buen funcionamiento de la generalización, el riesgo empírico, el cociente entre la dimensión de VC y el número de los puntos de referencias tienen que ser pequeños.

Puesto que el riesgo empírico es generalmente una función que disminuye *h*, resulta que para un número de muestras dadas, hay un valor óptimo de la dimensión de VC. La opción de un valor apropiado de *h* (que en la mayoría de las técnicas está controlado por el número de los parámetros libres del modelo) es muy importante para conseguir buenos funcionamientos, especialmente cuando el número de los puntos de referencias es pequeño.

Por lo tanto, una técnica, Minimización del Riesgo Estructural (MRE), fue desarrollada por Vapnik con el objetivo de resolver el problema de elegir una dimensión de VC apropiada. Además de un principio diferente de inducción, Principio Estructural de la Minimización del Riesgo.

Las Máquinas de Soporte Vectorial fueron desarrolladas para poner el principio de MRE en ejecución. Fueron utilizadas primeramente para la clasificación; también fueron aplicadas para solucionar problemas de regresión. Cuando las MSV fueron utilizadas para solucionar el problema de la regresión, generalmente fueron llamadas Maquinas de Soporte para la Regresión (MSR) y el objetivo es encontrar una función f con los parámetros **w** y **b** reduciendo al mínimo el riesgo de la regresión:

$$R_{reg}(f) = \frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle + C \sum_{i=1}^{N} l(f(\mathbf{x}_i), y_i),$$

Donde C es un término llamado coste del error, el primer término puede ser visto como el margen en la MSV y por lo tanto puede medir la dimensión de VC. Una interpretación común es que la norma euclidiana w mide la planicidad de la función f. Minimizando (w, w), la función objetivo será tan plana como sea posible.

La función f es definida como:

$$f(\mathbf{x}, \mathbf{w}, b) = \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}) \rangle + b,$$

### 1.11.2 Máquinas de Soporte para la Regresión (MSR) no lineal.

Para lograr una MSR no lineal se hace un mapeo de los patrones de entrada $x_i$ en algún espacio de características (en inglés, *feature space*) $\phi : \chi \to F$ Es suficiente conocer y usar $k(x, x') = \langle \phi(x), \phi(x') \rangle$ en vez de $\phi(.)$ explícitamente. Es decir hacer uso de una función kernel, para transformar el espacio de entrada. Esto nos permite plantear la solución como sigue:

$$\text{maximizar} \quad -\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(x_i, x_j) - \varepsilon\sum_{i=1}^{l}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*)$$

$$\text{sujetoa} \quad \left\{ \begin{array}{l} \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0 \\ \alpha_i, \alpha_i^* \in [0, C] \end{array} \right\}$$

La expansión de $f$ puede escribirse ahora como

$w = \sum_{i=1}^{l}(\alpha_i^* - \alpha_i)\phi(x_i)$ y por tanto

$$f(x) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)k(x_i, x) + b$$

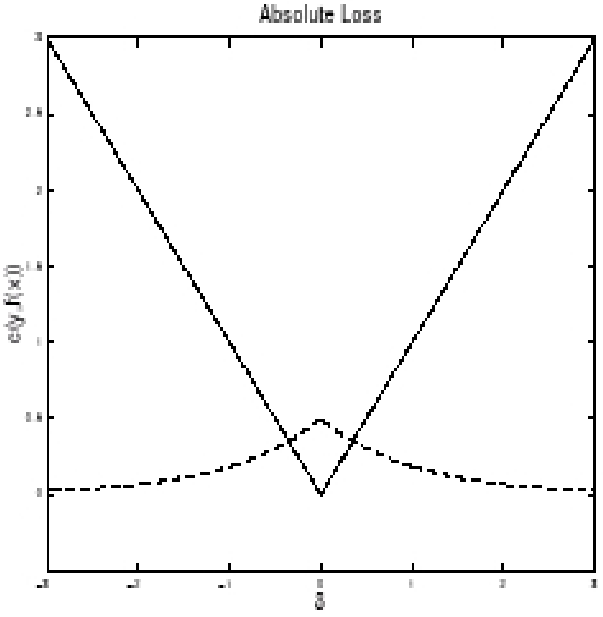
En el planteamiento no lineal el problema de optimización corresponde a encontrar la función más plana en el espacio de características, en vez de en el espacio de entrada.

### 1.10.3 Función de Pérdida.

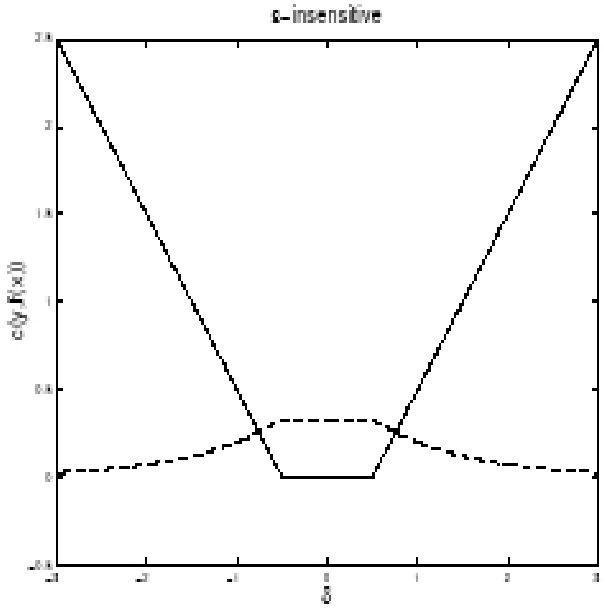
Para medir el riesgo empírico, debemos especificar una función de pérdida. Hay muchas funciones de pérdida, por ejemplo, función ajustada de  pérdida, función de pérdida de Huber, función ε-insensible.

| | loss function $l(\delta)$ | density function $p(\delta)$ |
|---|---|---|
| Linear $\varepsilon$-insensitive | $\lvert\delta\rvert_\varepsilon$ | $\frac{1}{2(1+\varepsilon)}\exp(-\lvert\delta\rvert_\varepsilon)$ |
| Laplacian | $\lvert\delta\rvert$ | $\frac{1}{2}\exp(-\lvert\delta\rvert)$ |
| Gaussian | $\frac{1}{2}\delta^2$ | $\frac{1}{\sqrt{2\pi}}\exp(-\frac{\delta^2}{2})$ |
| Quadratic $\varepsilon$-insensitive | $\frac{1}{2}\delta_\varepsilon^2$ | $\frac{1}{\lambda^*}\exp(-\frac{1}{2}\delta_\varepsilon^2)$<br>$\lambda^*=\varepsilon^2\sqrt{2\pi}-\sqrt{2\pi(\varepsilon^2\varepsilon-1)}+2\sqrt{2\varepsilon}$ |
| Huber's robust | $\begin{cases}\frac{1}{2\sigma}\delta^2, & \text{if } \lvert\delta\rvert\le\sigma\\ \lvert\delta\rvert-\frac{\sigma}{2}, & \text{otherwise}\end{cases}$ | $\propto\begin{cases}\exp(-\frac{1}{2\sigma}\delta^2), & \text{if } \lvert\delta\rvert\le\sigma\\ \exp(\frac{\sigma}{2}-\lvert\delta\rvert), & \text{otherwise}\end{cases}$ |
| Polynomial | $\frac{1}{d}\lvert\delta\rvert^d$ | $\frac{d}{2\Gamma(1/d)}\exp(-\lvert\delta\rvert^d)$ |
| Piecewise polynomial | $\begin{cases}\frac{1}{d\sigma^{d-1}}\delta^d, & \text{if } \lvert\delta\rvert\le\sigma\\ \lvert\delta\rvert-\sigma\frac{d-1}{d}, & \text{otherwise}\end{cases}$ | $\propto\begin{cases}\exp(-\frac{\lvert\delta\rvert^d}{d\sigma^{d-1}}), & \text{if } \lvert\delta\rvert\le\sigma\\ \exp(\sigma\frac{d-1}{d}-\lvert\delta\rvert), & \text{otherwise}\end{cases}$ |

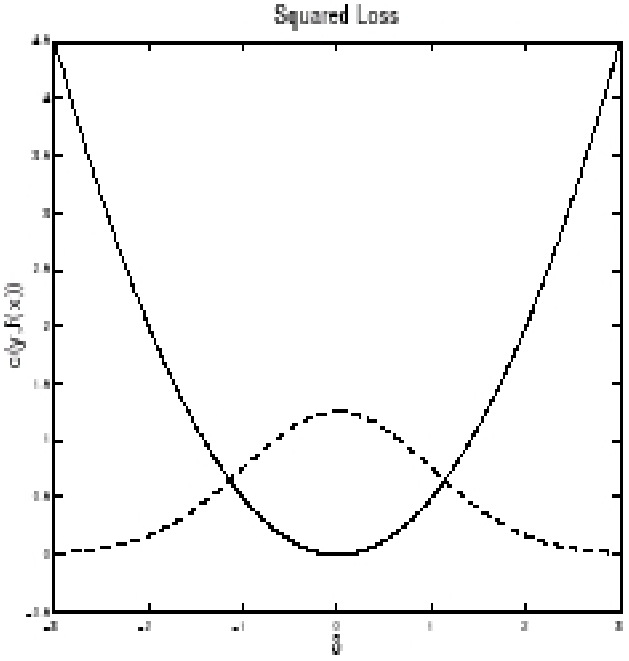
**Fig. 1.9:** Funciones de  pérdida y sus funciones correspondientes de la densidad.



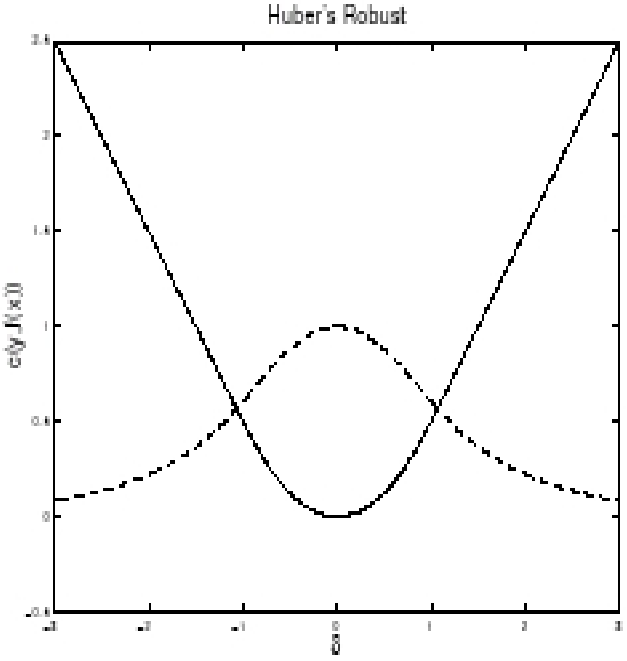

(a) Laplacian




(b) ε-insensitive




(c) Gaussian




(d) Huber's Robust

La función ε-insensible cuadrática de pérdida es la resultante de las funciones enumeradas.

Hay una diferencia importante entre aproximar una función a partir de datos dados y/o estimar esa función a partir del conjunto de datos. En el primer caso, los datos son correctos, es decir, las medidas fueron tomadas sin ruido y lo que se quiere es obtener una función que asuma los mismos valores (hasta una precisión dada) de las medidas, esto está relacionado muy de cerca con los problemas estándares de interpolación. En los problemas de estimación de funciones, no se puede confiar completamente en los datos, pueden generarse con algún ruido adicional. En la mayoría de los casos no se necesita conocer incluso el modelo del ruido, sino tener solamente una pista de a qué se asemeja. Estos dos problemas pueden ser tratados con una misma formulación, la diferencia entre ellos radicará en la elección de la función de costo apropiada.

Los métodos para estimar dependencias funcionales basadas en datos empíricos tienen una larga historia. Fueron introducidos por grandes matemáticos: Gauss (1777-1855) y Laplace (1749-1827), los cuales sugirieron dos métodos distintos para estimar dependencias a partir de mediciones obtenidas en astronomía y física.

Gauss propuso el método de los mínimos cuadrados (LSM), mientras Laplace propuso el método de los mínimos módulos (LMM). Durante el siglo IXX y a principios del XX hubo preferencia por el método de los mínimos cuadrados: La solución con este método para funciones lineales tiene una forma compacta. También se probó que entre estimadores lineales e imparciales, el LSM era el mejor. Luego, en la segunda mitad del siglo XX, se notó que en muchas situaciones el conjunto de estimadores lineales e imparciales era demasiado estrecho para estar seguro que el mejor estimador en este conjunto era realmente bueno (es muy probable que el conjunto completo contenga solo estimadores “malos”).

En 1920 R.Fisher descubrió el método de máxima verosimilitud (ML) e introdujo un modelo para medidas con ruido aditivo.

De acuerdo con este modelo el resultado de la función $f(x,\alpha_0)$ en cualquier punto $x^*$ está alterada por un ruido aditivo (descrito por una densidad simétrica conocida $p_0(\xi)$; $\xi$ no está correlacionada con $x^*$).

En el conjunto de estimadores imparciales, no necesariamente lineales, el método ML alcanza la menor varianza (estimador imparcial *efectivo*). Esto implica que si el ruido sigue una distribución Gaussiana (normal), entonces el LSM da la mejor solución, sin embargo, si el ruido está definido por una distribución de Laplace

$$p(x,\Delta) = \frac{1}{2\Delta}\exp\left\{-\frac{|\xi|}{\Delta}\right\}$$

la mejor solución es obtenida con el estimador de mínimos módulos. De estos resultados se infiere que la función de costo para el mejor (*efectivo*) estimador es definida por la distribución del ruido.

Generalmente la forma del ruido es desconocida. En 1960 Tukey demostró que en situaciones reales la forma del ruido está lejana tanto de las leyes Gaussianas como Laplacianas. Por tanto, se convirtió en una tarea importante crear una buena estrategia para estimar funciones en situaciones reales (cuando la forma del ruido es desconocida). Esta estrategia fue sugerida por P. Huber, quien creó el concepto de estimadores robustos.

Considerando la clase $H$ de densidades formadas por mezclas:

$$p(\xi) = (1-\varepsilon)g(\xi) + \varepsilon h(\xi)$$

de una cierta densidad fija $g(\xi)$ y una densidad arbitraria $h(\xi)$ donde ambas densidades son simétricas con respecto al origen. El peso en la mixtura es $1-\varepsilon$ y $\varepsilon$ respectivamente.

La función de costo $\varepsilon$-insensible le aporta una propiedad nueva a las soluciones que dan las máquinas de soporte vectorial, llamada ¨escasez¨ (sparsity) de la solución.

Esta función tiene la ventaja de que no se necesitan todos los patrones de entradas para describir el vector de regresión w, disminuyendo el costo computacional.

La MSR debe predecir la propiedad biológica de una muestra de entrenamiento a partir de rasgos estructurales de las moléculas, con una calidad que permita predecir la actividad de una muestra externa.

La e-insensible es la función de pérdida de mayor robustez y de mejores resultados en la detección de los mínimos cuadrados que ha sido reportada, la cual permite que se mantenga la generalización de las soluciones que dan las MSV.

Por tanto, se puede afirmar que en este tipo de técnicas su estructura se determina sobre la base del conjunto de entrenamiento, necesitándose pocos parámetros para el mismo. El entrenamiento se reduce a la solución de un problema de optimización de aquella función Kernel capaz de extraer la mayor cantidad de información de los datos relacionada con la propiedad, lo que al mismo tiempo reduce el problema de programación cuadrática, por lo que su solución es completamente reproducible. Al mismo tiempo, el uso de las funciones Kernels muestra mayor eficiencia en el manejo de los datos. Este procedimiento se emplea tanto para la generación de modelos a partir de la descripción de la estructura por fragmentos como por descriptores moleculares.

## 1.11 Kernel

Como se ha visto a lo largo del capítulo, la mayor parte de los conjuntos de datos dentro de la ingeniería química son no lineales, y que las MSV se basan en el aprendizaje lineal con un gran margen para el ajuste de los datos. Por lo tanto, es necesario tener algunas técnicas de muestreo no lineales para que la mayor parte de los conjuntos de datos en la química o la ingeniería química sen convenientemente tratados con MSV. Las funciones Kernel son las herramientas eficaces para este propósito.

La idea de una función kernel es lograr las transformaciones necesarias en el espacio de entrada sin tener que operar directamente en un espacio de características de mayor dimensionalidad. Por tanto los productos internos no necesitan ser evaluados en este espacio de mayor dimensión.

Toda esta teoría está basada en la reproducción de kernels en el espacio de Hilbert (RKHS del inglés Reproducing Kernel Hilbert Spaces) (Aronszajn,1950; Girosi, 1997; Heckman, 1997; Wahba, 1990). Un producto interno en el espacio de características tiene un kernel equivalente en el espacio de entrada,

$$K(x, x') = \langle \phi(x), \phi(x') \rangle \quad ,$$

provisto de ciertas condiciones. Si K es una función simétrica definida positiva, la cual satisface las condiciones de Mercer,

$$K(x,x') = \sum_{m}^{\infty} a_m \phi_m(x)\phi_m(x'), \quad a_m \geq 0,$$

$$\iint K(x,x')g(x)g(x')dxdx' > 0, \quad g \in L_2,$$

entonces el kernel representa un producto interno legítimo en el espacio de características.

El problema de la selección de un tipo de kernel para un problema en particular es una de las tareas más difíciles dentro de las máquinas de soporte vectorial.

## 1.12 Aplicaciones de las MSV

Las MSV atraen cada vez más a los científicos e ingenieros para modelar problemas de la vida real debido a su fundamentación teórica y avanzado funcionamiento probado en muchos casos.

Comparadas con otros métodos de procesamiento de datos, las MSV son especialmente convenientes para solucionar problemas del tamaño de muestra pequeño, con funcionamiento superior en la predicción.

Se muestran algunos de los ejemplos del uso de las MSV en el diseño de trabajos, materiales y experimentos de QSAR/QSPR, modelado para el control óptimo en la industria química, y otros ramas en tecnología de la química.

Muchas han sido las personas que han utilizado MSV para dar soluciones a diferentes problemas, específicamente en el trabajo con moléculas a través de descriptores. Por ejemplo:

1. Las MSV son utilizadas para desarrollar los modelos QSAR que correlacionan las estructuras moleculares a su toxicidad y bioactividades. El funcionamiento y la capacidad predictiva de MSV se investigan y se comparan con otros métodos tales como regresión lineal múltiple y métodos de redes neuronales.

En ese estudio, dos diversos juegos de datos se evaluaron**.** El primero implica el uso de MSV para el desarrollo de un modelo QSAR para la predicción de toxicidades de 153 fenoles, y la segunda se ocupa del modelo QSAR entre las estructuras y las actividades de un sistema de 85 inhibidores de cyclooxygenase-2 (COX-2). Para cada uso, las estructuras moleculares se describieron usando parámetros fisicoquímicos o descriptores moleculares.

En ambos casos, la capacidad predictiva del modelo de MSV es comparable o superior a ésos obtenidos por MLR y RBFNN. Los resultados indican que las MSV se pueden utilizar como alternativas para los estudios de QSAR.

2. Una selección de datos de 146 productos químicos naturales, sintéticos y ambientales pertenecientes a una amplia gama de familias de compuestos se evaluaron por su afinidad al receptor del androgénico. Estos productos químicos llamados comúnmente compuestos de interrupción endocrina (EDCs) presentan una variedad de efectos nocivos en seres humanos y animales. En ese trabajo, se determinaron las relaciones QSAR usando tres métodos, la regresión lineal múltiple (MLR), las redes neuronales (RBFNN) y las MSV. Cinco descriptores, enlaces por puente de hidrogeno en la interacción, la distribución de cargas atómicas y grado de ramificación molecular, se seleccionaron a partir de un método heurístico para construir modelos QSAR predictivos. La comparación de los resultados obtenidos a partir de los tres modelos, demostró que el método de MSV exhibió los mejores modelos, con un error cuadrático medio del 0.54 para el sistema del entrenamiento, 0.59 para el sistema de prueba, y 0.55 para la muestra completa.

3. La MSV, se utilizó para desarrollar un modelo QSPR de la energía en enlace de la disociación del O-H (BDE) de 78 fenoles substituidos. Los seis descriptores calculaban únicamente las estructuras moleculares de los compuestos seleccionados por la regresión “foreward stepwise” utilizando solamente las entradas para la MSV. El error cuadrático medio en las predicciones de los BDE para el entrenamiento, la prueba, y los conjuntos de datos totales fueron 3.808, 3.320, y 3.713 unidades de los BDE (kJ mol (- 1)), respectivamente.

Los resultados obtenidos por el núcleo Gaussiano de la MSV fueron mejores que los obtenidos por la regresión lineal múltiple, por las redes neuronales con función de base radial, el núcleo lineal de la MSV, y otros enfoques QSPR.

## 1.13 Tendencias y Tecnologías utilizadas

¿Qué metodología se debe usar para el desarrollo de esta aplicación?

Todo desarrollo de software es riesgoso y difícil de controlar por lo que se hace necesario la utilización de una metodología apropiada para el caso en cuestión. Cuando el proyecto que se va a desarrollar es de mayor envergadura, se hace necesaria la utilización de una metodología de desarrollo. Lo cierto es que muchas veces no se encuentra la más adecuada y se opta por hacer o diseñar una metodología propia, algo que por supuesto no esta mal, siempre y cuando cumpla con el objetivo.

Existen diferentes tipos de metodologías de desarrollo, las más conocidas y utilizadas son: RUP, XP, Extreme Programing (XP), FDD y MSF.

### 1.13.1 Proceso Unificado de Desarrollo de Software (Rational Unified Process (RUP))

Teniendo en cuenta las características del proyecto se utilizó como metodología de desarrollo RUP, ya que garantiza la elaboración de todas las fases de un producto de software orientado a objeto. RUP utiliza UML, que es un lenguaje que permite la modelación de sistemas con tecnología orientada a objetos, permite completar una gran parte de las disciplinas (flujos fundamentales) del proceso unificado de Rational (RUP):

- Modelado del negocio.
- Captura de requisitos (parcial).
- Análisis y diseño (completo).
- Implementación (como ayuda).
- Control de cambios y gestión de configuración (parte).
- Características Principales.
- Admite como notaciones: UML, COM, OMT y Booch.
- Realiza Chequeo semántico de los modelos.
- Ingeniería “de ida y vuelta”: Rose permite generar código a partir de modelos y viceversa.
- Desarrollo multiusuario.
- Integración con modelado de datos.
- Generación de documentación.

- Tiene un lenguaje de script para poder ampliar su funcionalidad.

La metodología RUP, llamada así por sus siglas en inglés, divide en 4 fases el desarrollo del software.

**Inicio:** El objetivo en esta etapa es determinar la visión del proyecto.

**Elaboración:** En esta etapa el objetivo es determinar la arquitectura óptima.

**Construcción:** En esta etapa el objetivo es obtener la capacidad operacional inicial.

**Transición:** El objetivo es llegar a obtener el release del proyecto.

Cada una de estas etapas es desarrollada mediante el ciclo de iteraciones, lo cual consiste en reproducir el ciclo de vida en cascada a menor escala. Los objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes.

Las características principales del proceso son:

- Guiado por los Casos de Uso.
- Centrado en la Arquitectura.
- Iterativo e Incremental.

RUP implementa las siguientes mejores prácticas asociadas al proceso de Ingeniería de Software:

- Desarrollo Iterativo.
- Manejo de los Requerimientos.
- Uso de una Arquitectura basada en componentes.
- Modelización visual.
- Verificación continua de la calidad.
- Manejo de los Cambios.

Puntos claves de RUP:

- Pesado.
- Dividido en cuatro fases.
- Las fases se dividen en iteraciones.
- El discurrir del proyecto se define en Workflows.
- Los artefactos son el objetivo de cada actividad.
- Se basa en roles.

### 1.13.2 Lenguaje representativo UML (Unified Modeling Language)

UML es una consolidación de muchas de las notaciones y conceptos más usados orientados a objetos. Una técnica para la especificación de sistemas en todas sus fases. Nació en 1994 cubriendo los aspectos principales de todos los métodos de diseño antecesores y, precisamente, los padres de UML son Grady Booch, autor del método Booch; James Rumbaugh, autor del método OMT e Ivar Jacobson, autor de los métodos OOSE y Objectory. La versión 1.0 de UML fue liberada en Enero de 1997 y ha sido utilizado con éxito en sistemas construidos para toda clase de industrias alrededor del mundo: hospitales, bancos, comunicaciones, aeronáutica, finanzas, etc.

Es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema software orientado a objetos. Uno de los objetivos principales de la creación de UML era posibilitar el intercambio de modelos entre las distintas herramientas CASE orientadas a objetos del mercado. Para ello era necesario definir una notación y semántica común. Hay que tener en cuenta que el estándar UML no define un proceso de desarrollo específico, tan solo se trata de una notación.

Se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real. UML ofrece nueve diagramas en los cuales modelar sistemas.

- Diagramas de Casos de Uso para modelar los procesos 'business'.
- Diagramas de Secuencia para modelar el paso de mensajes entre objetos.
- Diagramas de Colaboración para modelar interacciones entre objetos.
- Diagramas de Estado para modelar el comportamiento de los objetos en el sistema.
- Diagramas de Actividades para modelar el comportamiento de los Casos de Uso, objetos u operaciones.
- Diagramas de Clases para modelar la estructura estática de las clases en el sistema.
- Diagramas de Objetos para modelar la estructura estática de los objetos en el sistema.
- Diagramas de Componentes para modelar componentes.
- Diagramas de Implementación para modelar la distribución del sistema.

Los principales beneficios de son:

- Mejores tiempos totales de desarrollo (de 50 % o más).
- Modelar sistemas (y no sólo de software) utilizando conceptos orientados a objetos.
- Establecer conceptos y artefactos ejecutables.
- Encaminar el desarrollo del escalamiento en sistemas complejos de misión crítica.
- Crear un lenguaje de modelado utilizado tanto por humanos como por máquinas.
- Mejor soporte a la planeación y al control de proyectos.
- Alta reutilización y minimización de costos.
- Muy organizativo.
- Mucha documentación.

### 1.13.3 Herramientas Case. Visual Paradigm

Visual Paradigm es una herramienta que sirve para realizar modelado UML siguiendo el estándar UML 2.1. Tiene características gráficas muy cómodas que facilitan la realización de los diagramas de modelado que sigue el estándar de UML como son:

- Diagramas de Clase, Casos de Uso, Comunicación, Secuencia, Estado, Actividad, Componentes, etc.

Entre otras características importantes tenemos:

- Integración con diversas IDE's como son:
  - NetBeans (de Sun)
  - JDeveloper (de Oracle)
  - Eclipse (de IBM)
  - JBuilder (de Borland)
- Ingeniería Inversa para:

  - JAVA
  - .NET
  - XML
  - Hibernate

- Exportación de imágenes jpg, png y svg (w3g estándar).
- Trabaja sobre sistema operativo Linux.

### 1.13.4 Lenguaje de programación JAVA.

Java posee una curva de aprendizaje muy rápida. Resulta relativamente sencillo escribir applets interesantes desde el principio.

#### Orientado a objetos

Java fue diseñado como un lenguaje orientado a objetos desde el principio. Los objetos agrupan en estructuras encapsuladas tanto sus datos como los métodos (o funciones) que manipulan esos datos. La tendencia del futuro, a la que Java se suma, apunta hacia la programación orientada a objetos, especialmente en entornos cada vez más complejos y basados en red.

#### Distribuido

Java proporciona una colección de clases para su uso en aplicaciones de red, que permiten abrir sockets y establecer y aceptar conexiones con servidores o clientes remotos, facilitando así la creación de aplicaciones distribuidas.

#### Robusto

Java fue diseñado para crear software altamente fiable. Para ello proporciona numerosas comprobaciones en compilación y en tiempo de ejecución. Sus características de memoria liberan a los programadores de una familia entera de errores (la aritmética de punteros), ya que se ha prescindido por completo de los punteros, y la recolección de basura elimina la necesidad de liberación explícita de memoria.

#### Indiferente a la arquitectura

Java está diseñado para soportar aplicaciones que serán ejecutadas en los más variados entornos de red, desde Unix a Windows Nt, pasando por Mac y estaciones de trabajo, sobre arquitecturas distintas y con sistemas operativos diversos. Para acomodar requisitos de ejecución, el compilador de Java genera bytecodes: un formato intermedio indiferente a la arquitectura diseñado para transportar el código eficientemente a múltiples plataformas hardware y software. El resto de los problemas los soluciona el intérprete de Java.

### 1.13.5 Ambientes de desarrollo

#### Eclipse

La plataforma Eclipse, combinada con el JDT (*Java Development Tooling*), permite disponer de un IDE (*Integrated Development Environment* o Entorno de desarrollo integrado) para Java de excelente calidad.
Como IDE de Java, Eclipse posee un editor visual con sintaxis coloreada, ofrece compilación incremental de código, un potente depurador (que permite establecer puntos de interrupción, modificar e inspeccionar valores de variables, e incluso depurar código que resida en una máquina remota), un navegador de clases, un gestor de archivos y proyectos, pero no se limita sólo a esto. Es muy eficiente para reducir los tiempos de depuración y pruebas.
La versión estándar de Eclipse proporciona también una biblioteca de refactorización de código y una lista de tareas, soporta la integración con JUnit, y suministra un *front-end* gráfico para Ant, la conocida herramienta de código abierto que forma parte del proyecto Jakarta de Apache.
También incluye una herramienta para completar código: el asistente de contenido, encargado de mostrar los métodos y atributos de las clases con las que se está trabajando, ya formen parte de las APIs de Java o de cualquier otra clase en el *build path*, aunque estén en ficheros JAR.
Este asistente también proporciona información de cada uno de los métodos mediante una ventana secundaria contextual, y avisa cuándo se graba la clase o la interfaz de los errores cometidos al escribir el código. Eclipse incluye también asistentes para la creación de clases e interfaces, y proporciona una integración perfecta con el sistema open source CVS (*Concurrent Version System*), muy útil para llevar el control de las versiones con las que se trabaja y conocer en todo momento sus respectivas diferencias.

Eclipse al igual que Visual Age for Java no cuenta con un menú de compilación pues no es necesario: cada vez que se hacen cambios en uno o más ficheros, el compilador interno de Eclipse recompila todos los ficheros fuente afectados por los cambios. El usuario no tiene que preocuparse de compilar, y puede estar seguro de contar con archivos siempre compilados. En consecuencia, tampoco resulta preciso esperar a la compilación para detectar ciertos errores: Eclipse muestra indicaciones de los errores aparecidos según se van realizando o guardando los cambios.

### 1.13.6 Sistema gestor de bases de datos.

MySQL es un sistema de administración de bases de datos relacional (RDBMS). Se trata de un programa capaz de almacenar gran cantidad de datos y distribuirlos para cubrir las necesidades de cualquier tipo de organización, desde pequeños establecimientos comerciales a grandes empresas y organismos administrativos. Es una aplicación de código abierto que permite redistribuir una aplicación que la contenga y modificar su código para mejorarla o adaptarla a nuestras necesidades. Además, existe la seguridad de contar con una importante cuota de mercado y de saber que es una solución estable, mantenida por un buen equipo de desarrolladores y e incluso con soporte de pago. Compite con sistemas RDBMS como Oracle, SQL Server entre otros. Incluye todos los elementos necesarios para instalar el programa, preparar diferentes niveles de acceso de usuarios, administrar el sistema y proteger los datos. Utiliza el lenguaje de consulta estructurado.

### MySQL 5.0.x

Finalmente como sistema gestor de bases de datos se ha utilizado MySQL en su versión 5.0.

- Uno de los gestores más rápidos que se encuentran en el mercado.
- Compatibilidad entre sus versiones en diferentes arquitecturas.
- Muy fácil de usar.
- Se obtiene gratis o a bajo costo.

## Conclusiones

En este capítulo se profundizó en el conocimiento de algunos conceptos necesarios para la compresión de este trabajo; lo cual permitió reconocer las MSV como la herramienta matemática con más aplicaciones en los últimos años en al campo de la química medicinal. Se realizó además un análisis completo de las tecnologías utilizadas a lo largo del desarrollo del sistema propuesto, y se fundamentaron las elecciones del lenguaje, el sistema gestor de bases de datos, y la metodología a utilizar. Se hizo también una descripción de las tendencias y tecnologías actuales.

# CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

## Introducción

En el presente capítulo se describe la solución propuesta utilizando los componentes del modelo de dominio de la metodología RUP. De este modelo se tiene en cuenta la definición de las entidades y los conceptos principales, así como su representación gráfica. Además se describen los requisitos funcionales y no funcionales del sistema, los actores que intervienen en el sistema, así como los diagramas de casos de uso.

Para el desarrollo de un software, RUP propone como primer paso, que se comprenda el contexto a automatizar, como fuente que aporta información importante para la obtención de los requerimientos que debe cumplir el sistema e identificar los actores del mismo. Con este fin se lleva a cabo la modelación del negocio, que permite una mejor comprensión de la dinámica y estructura organizacional, identificar los problemas actuales y posibles mejoras y derivar los requerimientos del sistema. Cuando no existen procesos definidos RUP propone realizar un modelo del dominio, que es un subconjunto del modelo de negocio.

## 2.1 Qué es un Modelo del Dominio

Un modelo del dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las "cosas" que existen o los eventos que suceden en el entorno en el que trabaja el sistema.

La modelación del dominio tiene como objetivo fundamental la comprensión y descripción de las clases más importantes en el sistema.

La concepción de la aplicación a desarrollar se basa en los requerimientos funcionales y no funcionales, a partir de los cuales se identifican las opciones del sistema, y se describen sus casos de uso.

## 2.2 Diagrama Modelo del Dominio

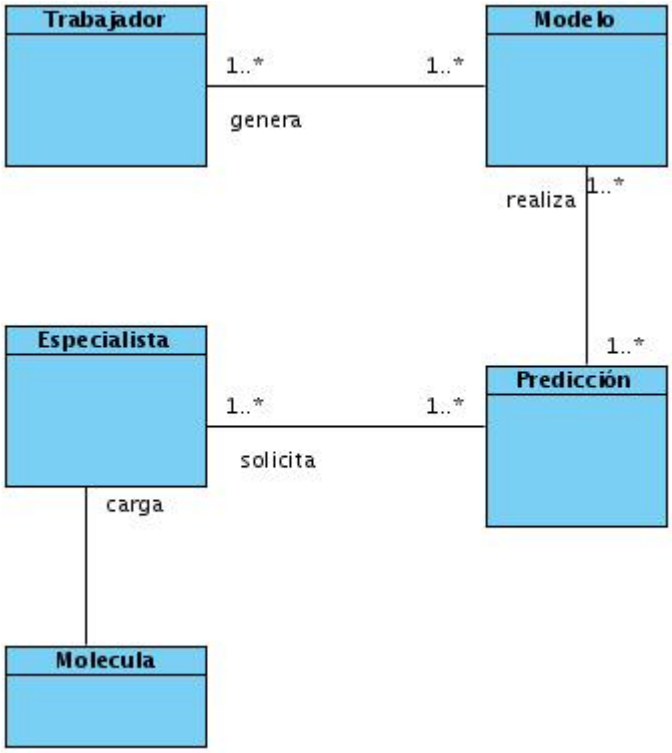



**Fig. 2.1**: Modelo del Dominio

## 2.3 Definición de los Requerimientos Funcionales

Un requerimiento funcional define el comportamiento interno del software: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que muestran cómo los casos de uso serán llevados a la práctica.

Permiten expresar una especificación más detallada de las responsabilidades del sistema. Son complementados por los requerimientos no funcionales.

La especificación de los requerimientos debe ser precisa, completa y clara. Con su definición se busca establecer un común entendimiento con el cliente sobre los objetivos del negocio propuesto, reflejan todo lo que el sistema debe hacer. A continuación se enumeran los requerimientos funcionales de la aplicación propuesta:

**RF1**: Realizar predicción.

**RF2**: Realizar entrenamiento.

**RF3**: Crear modelo.

**RF4**: Cargar Ficheros (de entrenamiento, de datos a predecir).

## 2.4 Definición de los Requerimientos No Funcionales

Los requerimientos no funcionales especifican propiedades del sistema como restricciones de ambiente y desarrollo, performance, dependencias de plataformas, mantenimiento y confiabilidad. Constituyen otros requisitos que no forman parte de la funcionalidad principal de la aplicación. Son propiedades o cualidades que el producto debe tener. Representan las características del producto.

Dentro de los requerimientos no funcionales para el desarrollo de la herramienta se encuentran:

- **Requisitos de funcionalidad**

  El sistema debe someterse a una etapa de adiestramiento en la que los usuarios se familiaricen con la aplicación y sean detectados los posibles errores, o puedan surgir posibles cambios en la interfaz para que los usuarios queden totalmente complacidos.

- **Apariencia o interfaz externa**

  El sistema debe contar con una interfaz amigable, fácil de comprender y usar, donde el usuario pueda orientarse fácilmente.

- **Usabilidad**

  El sistema le ofrecerá al usuario la posibilidad de realizar predicciones y analizar los resultados de las mismas, así como la creación de modelos para la predicción. En este sentido se centra el diseño de la aplicación y en específico de las interfaces que harán posible el intercambio de datos de manera que le resulte al usuario de fácil entendimiento.

- **Soporte**

  Terminada la aplicación será sometida a pruebas con el objetivo de comprobar su funcionalidad.

- **Requerimientos de Portabilidad**

  La herramienta propuesta podrá ser usada bajo cualquier sistema operativo, para su implementación se usaron Herramientas de Programación y Gestión de Bases de Datos que son multiplataforma.

- **Software**

  Se debe disponer de sistemas operativos Linux, Windows 95 o superior para la instalación de la aplicación. Debe tenerse instalado el Java Runtime Environment (JRE) versión 1.5 o superior.

- **Hardware**

  Para el desarrollo y puesta en práctica del proyecto se requieren máquinas con los siguientes requisitos:

  - Procesador Pentium III o superior
  - 256 Mb de RAM
  - 50 Mb de capacidad del disco duro

## 2.5 Actores del Sistema a Automatizar

Un actor es una entidad externa del sistema que de alguna manera participa en la historia del caso de uso. Por lo general estimula el sistema con eventos de entradas o recibe algo de él. O sea, es un rol de un usuario, que puede intercambiar información o puede ser un recipiente pasivo de información y representa a un ser humano, a un software o a una máquina que interactúa con el sistema. Acorde a la situación en la que se desarrolla la herramienta se encontraron los siguientes actores:

| Actores | Descripción |
|---|---|
| Especialista | Representa el usuario que va a interactuar con el sistema. |
| Trabajador | Es la persona encargada de actualizar el sistema y crear los modelos. |

**Tabla 2.1**: Actores del Sistema

## 2.6 Casos de Uso Definidos.

Los casos de uso se utilizan para obtener información de cómo debe trabajar el sistema, son descripciones de la funcionalidad del sistema independiente de la implementación. Describen bajo la forma de acciones y reacciones el comportamiento de un sistema desde el punto de vista del usuario.

Los casos de usos definidos son los siguientes:

**Crear modelo**

**Realizar predicción**

Un diagrama de casos de uso del sistema contiene actores, casos de uso del sistema y las relaciones entre ellos.

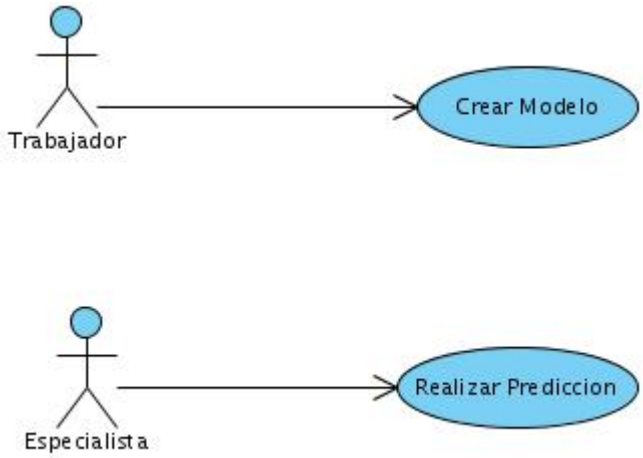



**Figura 2.2:** Diagrama de Casos de Uso del Sistema

## 2.7 Descripción de los Casos de Uso del Sistema

A continuación se presenta una descripción textual detallada del proceso en cada caso de uso del sistema.

| | |
|---|---|
| Caso de Uso: | Realizar Predicción. |
| Actores: | Especialista. |
| Propósito: | Predecir actividad biológica de la molécula a partir de descriptores, basado en máquinas de soporte vectorial. |
| Resumen: | El caso de uso se inicia cuando el especialista selecciona la opción predecir, carga un modelo, el fichero con los datos a predecir y solicita que se realice la predicción. El caso de uso concluye cuando el sistema guarda el resultado de la predicción en un fichero vacío que se le especifica al sistema. |
| Referencia: | **RF1, RF4.** |
| Precondiciones: | Que exista un modelo. |
| Poscondiciones: | Guardar en el fichero de salida el resultado de la predicción. |

| **Flujo Normal de Eventos** | |
|---|---|
| Acción del Actor | Respuesta del Sistema |
| 1. Escoge la opción predecir. | 1.1 Muestra las opciones para la predicción. |
| 2. Selecciona el modelo, carga en la aplicación el fichero con los datos a predecir y el fichero de salida. Oprime el botón Predecir. | 2.1 El sistema realiza la predicción y guarda en el fichero de salida los resultados de la predicción. |

| **Flujos Alternos** | |
|---|---|
| Acción del Actor | Respuesta del Sistema |

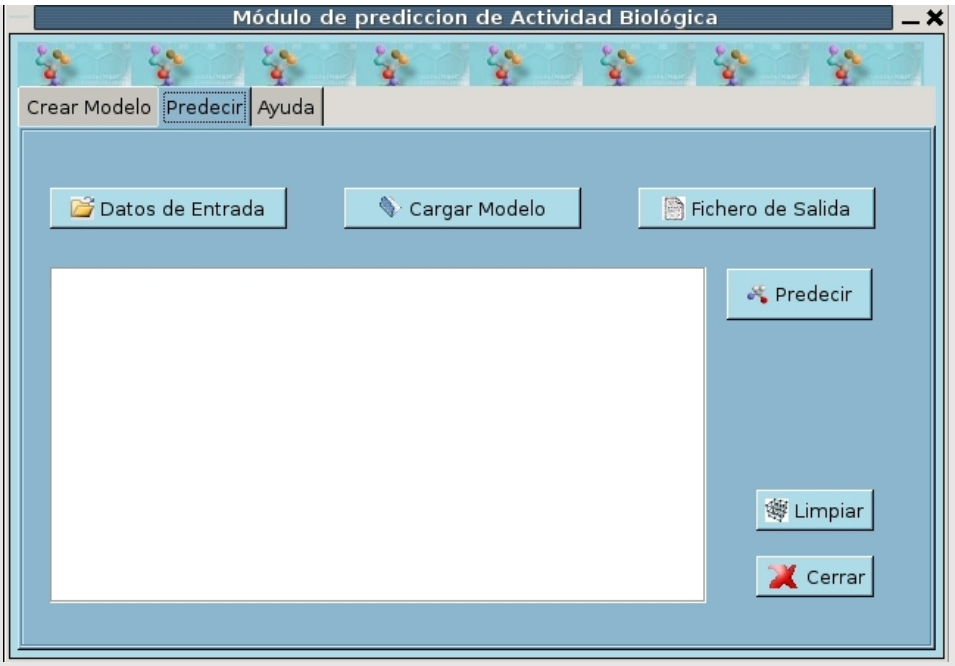
| | |
|---|---|
| Prototipo de Interfaz |  |
| Prioridad | Crítico |

**Tabla 2.2:** Descripción del Caso de Uso Realizar Predicción.

| Caso de Uso: | Crear Modelo. |
|---|---|
| Actores: | Administrador. |
| Propósito: | Crear modelos para la predicción de actividad biológica a partir de descriptores, basado en máquinas de soporte vectorial. |
| Resumen: | El caso de uso se inicia cuando el trabajador selecciona la opción Crear Modelo. Escoge los parámetros que tendrá la máquina de soporte vectorial que será entrenada y carga el fichero con los datos de entrenamiento. El caso de uso concluye cuando el administrador oprime el botón Crear Modelo, es aquí donde se entrena la máquina y se crea el modelo que es guardado automáticamente en un fichero. |
| Referencia: | **RF2, RF3, RF4.** |
| Precondiciones: | Que el fichero de entrenamiento exista. |
| Poscondiciones: | Crear modelo. |

| **Flujo Normal de Eventos** | |
|---|---|
| Acción del Actor | Respuesta del Sistema |
| 1. El administrador escoge la opción crear modelo. | 1.1 Muestra las opciones para crear el modelo |
| 2. Selecciona los parámetros para construir la máquina de soporte vectorial y carga el fichero de datos para el entrenamiento. Oprime el botón crear modelo. | 2.1 Construye y entrena la máquina de soporte vectorial y crea el modelo.<br>2.2 Guarda automáticamente el modelo en un fichero. |

| **Flujos Alternos** | |
|---|---|
| Acción del Actor | Respuesta del Sistema |

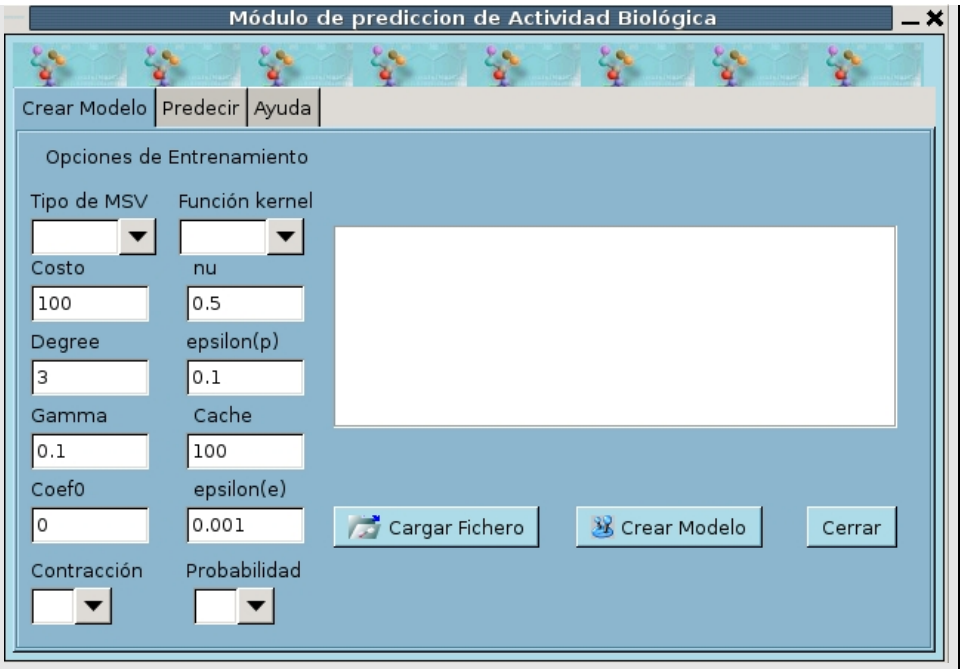
| | |
|---|---|
| Prototipo de Interfaz |  |
| Prioridad | Crítico |

**Tabla 2.3** Descripción del Caso de Uso Realizar Entrenamiento.

## Conclusiones

Debido a la poca estructuración de los procesos del negocio y para una mejor comprensión, se definieron en este capítulo conceptos, que fueron relacionados mediante un diagrama de Modelo del Dominio mostrando cómo se desarrolla este proceso. Se definieron además los requisitos que debe cumplir el sistema, los actores así como los casos de uso con los que contará el sistema y la descripción de cada uno de ellos.

# CAPÍTULO 3: ANÁLISIS Y DISEÑO

## Introducción

En el presente capítulo se dará una descripción del estilo arquitectónico utilizado para el desarrollo de la aplicación, se describirán los patrones de diseño empleados así como los diagramas de clases del análisis y del diseño.

## 3.1 Análisis y Diseño

El objetivo de este flujo de trabajo es traducir los requisitos a una especificación que describe cómo implementar el sistema.

Los objetivos del análisis y diseño son:

- Transformar los requisitos al diseño del futuro sistema.
- Desarrollar una arquitectura para el sistema.
- Adaptar el diseño para que sea consistente con el entorno de implementación, diseñando para el rendimiento.

El análisis consiste en obtener una visión del sistema, de modo que sólo se interesa por los requisitos funcionales. Específica el comportamiento funcional del sistema, independientemente de los aspectos relativos al ambiente en el que va a ser finalmente implementado. El modelo de análisis captura completamente y con exactitud los requerimientos del sistema.

Por otro lado el diseño es un refinamiento del análisis que tiene en cuenta los requisitos no funcionales, definiendo cómo el modelo de análisis orientado a la aplicación va a ser realizado en el ambiente de implementación, en definitiva cómo cumple el sistema sus objetivos.

El resultado final más importante de este flujo de trabajo será el modelo de diseño. Consiste en colaboraciones de clases, que pueden ser agregadas en paquetes y subsistemas.

### 3.2 Estilo arquitectónico utilizado.

La arquitectura en tres capas es, desde el punto de vista tecnológico, lo más avanzado que puede implementarse como estructura de las aplicaciones informáticas. Este modelo de tres capas consta de:

1. Capa de Presentación: aquí se colocan todos los componentes relacionados con la interfaz del usuario.
2. Capa de Negocio: también llamada lógica de negocio. Es la que realiza el procesamiento de la información, interactuando con el usuario mediante la capa de presentación, y con la base de datos mediante la capa de acceso a datos.
3. Capa de Acceso a Datos: por lo general implementada como una base de datos dentro de un Sistema de Administración de Bases de Datos Relacionales. Contiene, además de los datos en sí, código escrito en procedimientos almacenados, disparadores (triggers), reglas y funciones.

Las principales ventajas de este modelo son las siguientes:

1. Separación de funciones: todo lo relacionado con la interfaz del usuario va en una capa, las reglas de negocio en otra y el manejo de datos en una tercera capa. No se mezcla en una capa código correspondiente a otra.
2. Reutilización: el código correspondiente a una capa puede ser reutilizado desde varias partes de la capa inmediatamente superior.
3. Escalabilidad: sabiendo dónde está el código correspondiente a cada capa, pueden realizarse modificaciones dentro de una capa para mejorar o aumentar el tamaño del sistema de software, con un mínimo impacto en las capas restantes.
4. Facilidad de mantenimiento: mediante esta división, es mucho más sencillo localizar errores en el código o efectuar mejoras.

Para el desarrollo de la aplicación se utilizó el estilo arquitectónico Modelo Vista Controlador (Model-View-Controller - MVC), que es un tipo de arquitectura en tres capas. Este patrón de arquitectura de software separa los datos de una aplicación, de la interfaz del usuario y de la lógica de control de la aplicación en tres componentes distintos:

- Modelo: Esta es la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos. Es el responsable de acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento. Lleva un registro de las vistas y controladores del sistema.

- Vista: Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario. Las vistas son responsables de recibir datos del modelo y los muestra al usuario.
- Controlador: Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista. Es responsable de recibir los eventos de entrada (un clic, un cambio en un campo de texto, etc.). Contiene reglas de gestión de eventos, del tipo "SI Evento Z, entonces Acción W". Estas acciones pueden suponer peticiones al modelo o a las vistas.

### 3.2.1 Ventajas de MVC

• Soporte para múltiples vistas: ya que la vista se separa del modelo y no hay ninguna dependencia directa entre vista y modelo, la interfaz de usuario puede mostrar múltiples vistas de los mismos datos al mismo tiempo.

• Proporciona un mecanismo de configuración a componentes complejos muchos más tratable que el puramente basado en eventos (el modelo puede verse como una representación estructurada del estado de la interacción).

• Mayor soporte a los cambios: los requisitos de interfaz tienden a cambiar más rápidamente que las reglas de negocio. Puesto que el modelo no depende de las vistas, la adición de nuevos tipos de vista al sistema generalmente no afectan al modelo. Por tanto, el ámbito del cambio se limita a la vista.

Además de tener en cuenta sus ventajas, utilizamos el patrón de arquitectura Modelo Vista Controlador siguiendo el estándar utilizado para el desarrollo del proyecto.

## 3.3 Diagramas de Clases del Análisis

El diagrama de clases del análisis es la primera aproximación al diseño. En este diagrama se muestran las clases en función de sus responsabilidades (Clases Interfaz, Clases Controladoras y Clases Entidades) y la relación entre ellas. Se realizó un diagrama de clases del análisis para caso de uso (Realizar Predicción y Crear Modelo).

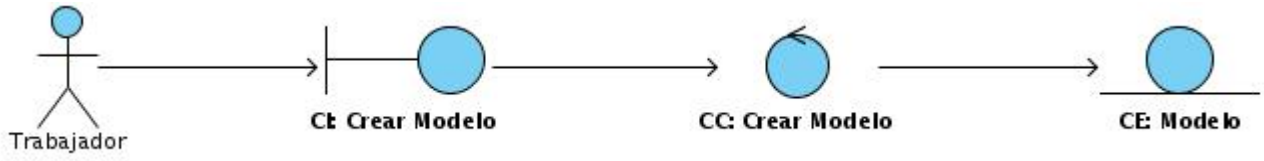



**Fig. 3.1** Diagrama de Clases del Análisis Crear Modelo.

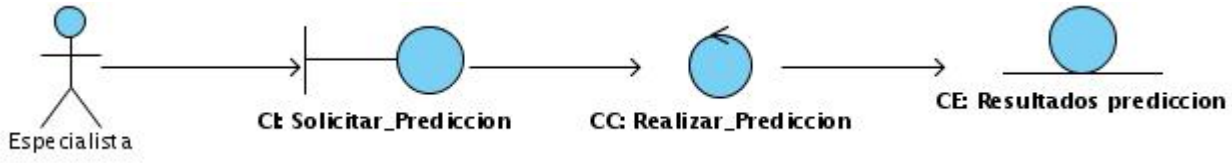



**Fig.3.2** Diagrama de Clases del Análisis Realizar Predicción.

## 3.4 Principales Patrones de Diseño utilizados

Los Patrones de Diseño (Design Patterns) son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

Un patrón de diseño es una solución a un problema de diseño no trivial que es efectiva (ya se resolvió el problema satisfactoriamente en ocasiones anteriores) y reusable (se puede aplicar a diferentes problemas de diseño en distintas circunstancias). Una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios.

Los patrones de diseño pretenden:

- Proporcionar catálogos de elementos reusables en el diseño de sistemas software.
- Evitar la reiteración en la búsqueda de soluciones a problemas ya conocidos y solucionados anteriormente.
- Formalizar un vocabulario común entre diseñadores.
- Estandarizar el modo en que se realiza el diseño.
- Facilitar el aprendizaje de las nuevas generaciones de diseñadores condensando conocimiento ya existente.

Asimismo, no pretenden:

- Imponer ciertas alternativas de diseño frente a otras.
- Eliminar la creatividad inherente al proceso de diseño.

En la realización del diseño para la aplicación informática a implementar, se utilizaron los patrones GRASP:

- **Experto**: Propone asignar la responsabilidad a la clase que cuenta con la información necesaria para cumplir la responsabilidad. Permitiendo que se conserve el encapsulamiento, soportando un bajo acoplamiento y una alta cohesión.
- **Creador**: Propone asignarle a una clase la responsabilidad de crear los objetos de la otra en los casos de: contener, agregar, registrar o utilizar. Brindando soporte de bajo acoplamiento, lo cual supone menos dependencias entre clases y posibilidades.

- **Bajo Acoplamiento**: brinda como solución asignar responsabilidades de manera que las clases no dependan fuertemente de otras. Ofreciendo como beneficio que son fáciles de entender por separadas, fáciles de reutilizar y no se afectan por cambios de otros componentes.
- **Alta Cohesión**: Este patrón propone asignar la responsabilidad de manera que la complejidad se mantenga dentro de límites manejables asumiendo solamente las responsabilidades que deben manejar, evadiendo un trabajo excesivo. Su utilización: mejora la claridad y facilidad con que se entiende el diseño, simplifica el mantenimiento y las mejoras de funcionalidad, generan un bajo acoplamiento, soporta mayor capacidad de reutilización.
- **Controlador**: Sugiere asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase que represente un manejador de los eventos del sistema. Este patrón ofrece mayor potencial de los componentes reutilizables garantizando que los procesos de dominio sean manejados por la capa de los objetos del dominio.

## 3.5 Diagrama de Clases del Diseño

Un diagrama de clases del diseño muestra las especificaciones para las clases software de una aplicación. Este tipo de diagramas muestra definiciones de entidades software más que conceptos del mundo real y se elabora para tener en cuenta los detalles concretos de la implementación del sistema. Es un modelo de objeto que describe la realización física de los casos de uso, que en este caso son dos: Realizar Predicción y Crear Modelo.

Incluye clases, asociaciones y atributos así como los métodos, especificados en el lenguaje de programación con el que se realizará la implementación y las relaciones y dependencias entre las clases. También se presentan los diagramas de interacción correspondientes a cada escenario por caso de uso, específicamente el de secuencia, que muestra las interacciones entre los objetos, ordenadas en secuencia temporal durante un escenario concreto.

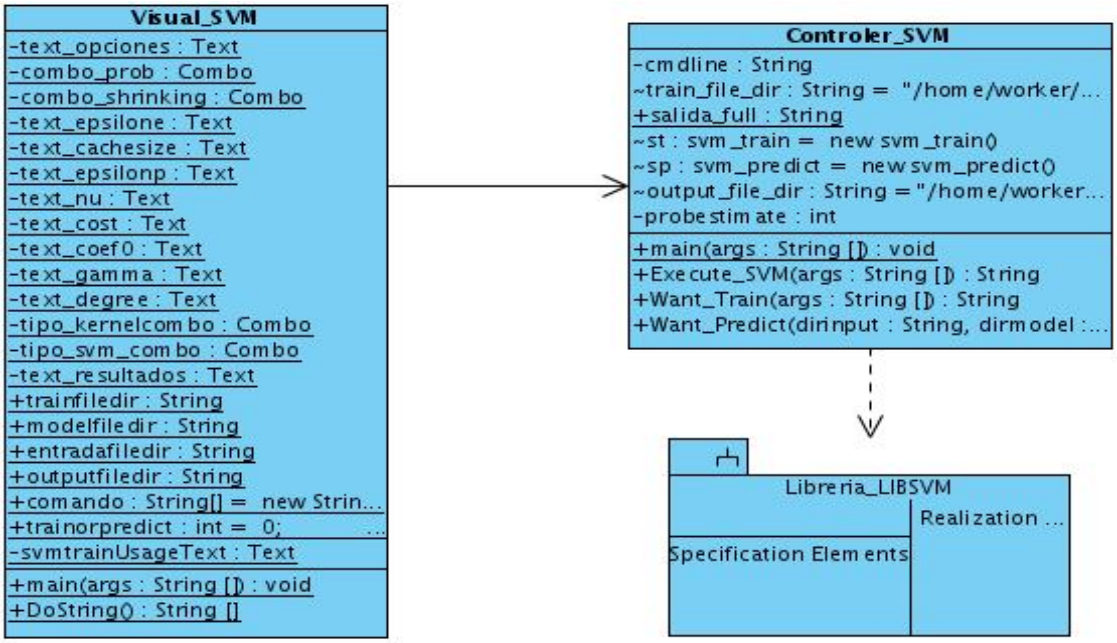



**Fig. 3.3** Diagrama de Clases del Diseño.

### 3.6 Diagramas de secuencia.

Un diagrama de secuencia muestra las interacciones entre objetos ordenadas en secuencia temporal. Muestra los objetos que se encuentran en el escenario y la secuencia de mensajes intercambiados entre los objetos para llevar a cabo la funcionalidad descrita por el escenario.

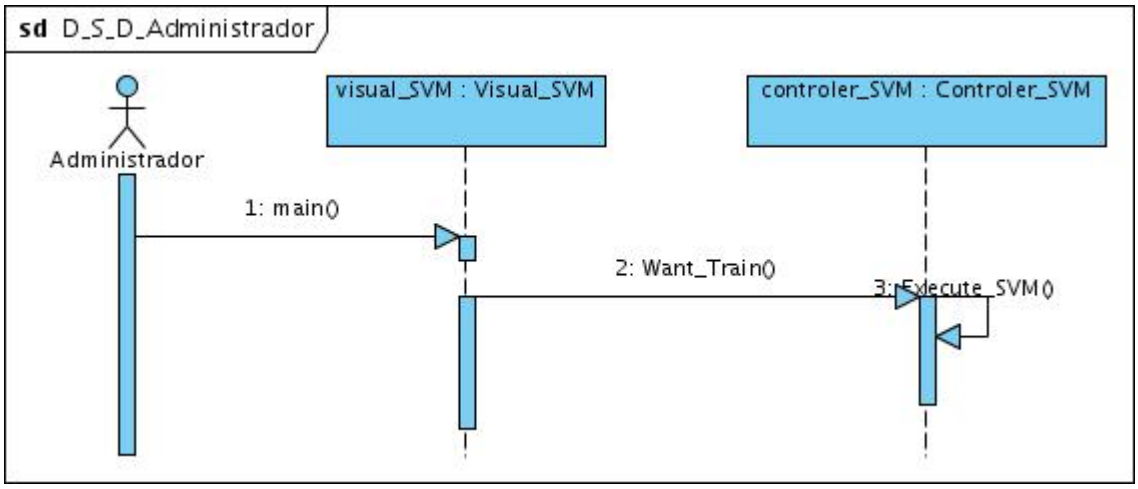



**Fig. 3.4** Diagrama de Secuencia Realizar Entrenamiento.

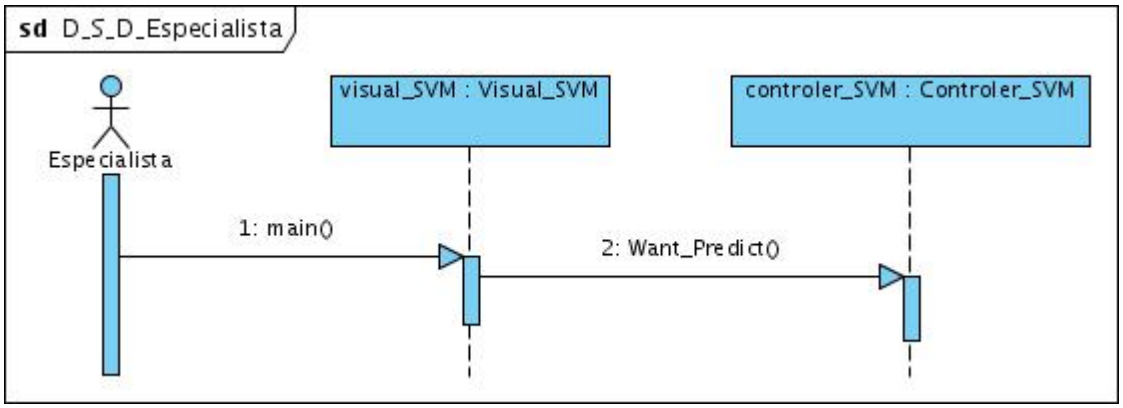



**Fig 3.5** Diagrama de Secuencia Realizar Predicción.

## Conclusiones

En el presente capítulo fue descrito el estilo arquitectónico utilizado en el diseño de la herramienta así como los patrones de diseño utilizados y la fundamentación de su uso. También fue desarrollado el diagrama de clases del análisis y del diseño.

# CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

En el presente capítulo se describe cómo los elementos del modelo de diseño se implementan en términos de componentes. Para esto se muestra el diagrama de componentes y se validan los modelos generados por la MSV. También se comparan los valores obtenidos con valores reales, con el objetivo de determinar el grado de precisión con que predicen los mismos.

## 4.1 Implementación

En este flujo de trabajo se implementan las clases y objetos en ficheros fuente, binarios y ejecutables. Además se muestran las pruebas para validar los modelos. El resultado final de este flujo de trabajo es un sistema ejecutable. La estructura de todos los elementos implementados forma el modelo de implementación. La disciplina de implementación limita su alcance a cómo las clases individuales serán probadas.

### 4.1.1 Diagrama de Componentes

Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes software (componentes de código fuente, binarios o ejecutables). Desde el punto de vista del diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión del software, la reutilización, y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo. A continuación se muestra el diagrama de componentes del software en cuestión:

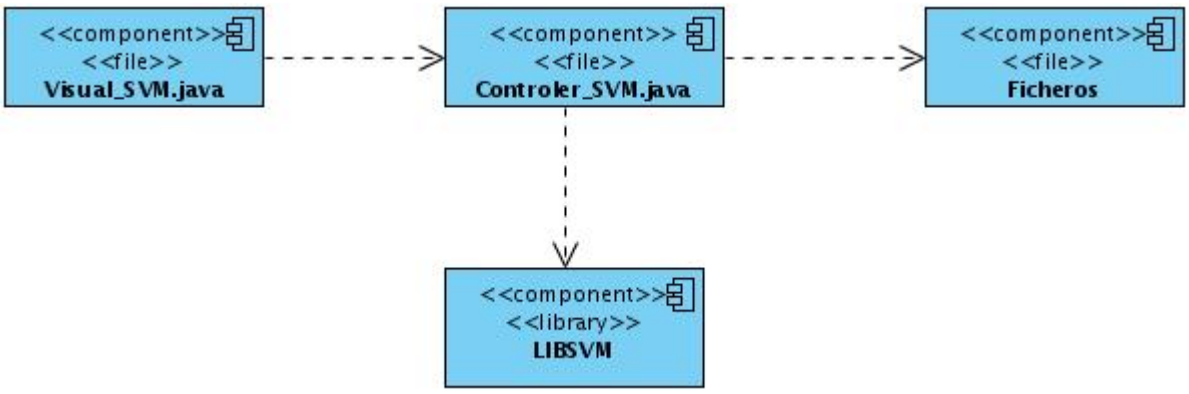



**Fig. 4.1** Diagrama de Componentes.

## 4.2 Validación de Modelos.

La prueba de software es un elemento crítico para la garantía de la calidad y representa una revisión final de las especificaciones del diseño y de la codificación. Las pruebas no pueden asegurar la ausencia de defectos; solo pueden demostrar que existen defectos en el software.

La construcción de modelos predictivos de actividad biológica consiste en la creación de un modelo (de clasificación o regresión) a partir de un conjunto de entrenamiento. Una vez que se ha construido el modelo se puede utilizar para predecir automáticamente muestras no clasificadas. La efectividad de los modelos creados depende, entre otros factores, de la naturaleza de los descriptores moleculares seleccionados para caracterizar la estructura química. Las variables independientes son los descriptores de Randic, Valencia y Partición de la Refractividad Molecular calculados a 2000 moléculas seleccionadas del ensayo 155 del NCI Yeast Anticancer Drug Scree respetando la proporción existente entre activos e inactivos, para que la muestra presente caraterísticas similares a la Base de Datos general, en una relación de 8X1 entre moléculas inactivas y activas. De la muestra se tomó el 80% para entrenar la MSV y crear el modelo y el 20% para realizar las pruebas. La muestra de prueba esta compuesta por 400 moléculas con una relación también de 8x1 entre moléculas inactivas y activas. Con la aplicación se generaron varios modelos de clasificación, con parámetros específicos en cada caso y se realizó la predicción con cada uno de ellos.

La creación de una MSV es un problema combinatorio, ya que en la aplicación existen 3 tipos de MSV para clasificación y cada una tiene la posibilidad de escoger uno de los 5 kernels existentes en la aplicación. Para las pruebas se escogió la MSV c_svc y como función kernel FBR, teniendo en cuenta que es la combinación más utilizada y la que mejor resultado mostró. Después de fijar los parámetros específicos del kernel (gamma) y el costo de la función, se creó el modelo para realizar la predicción.

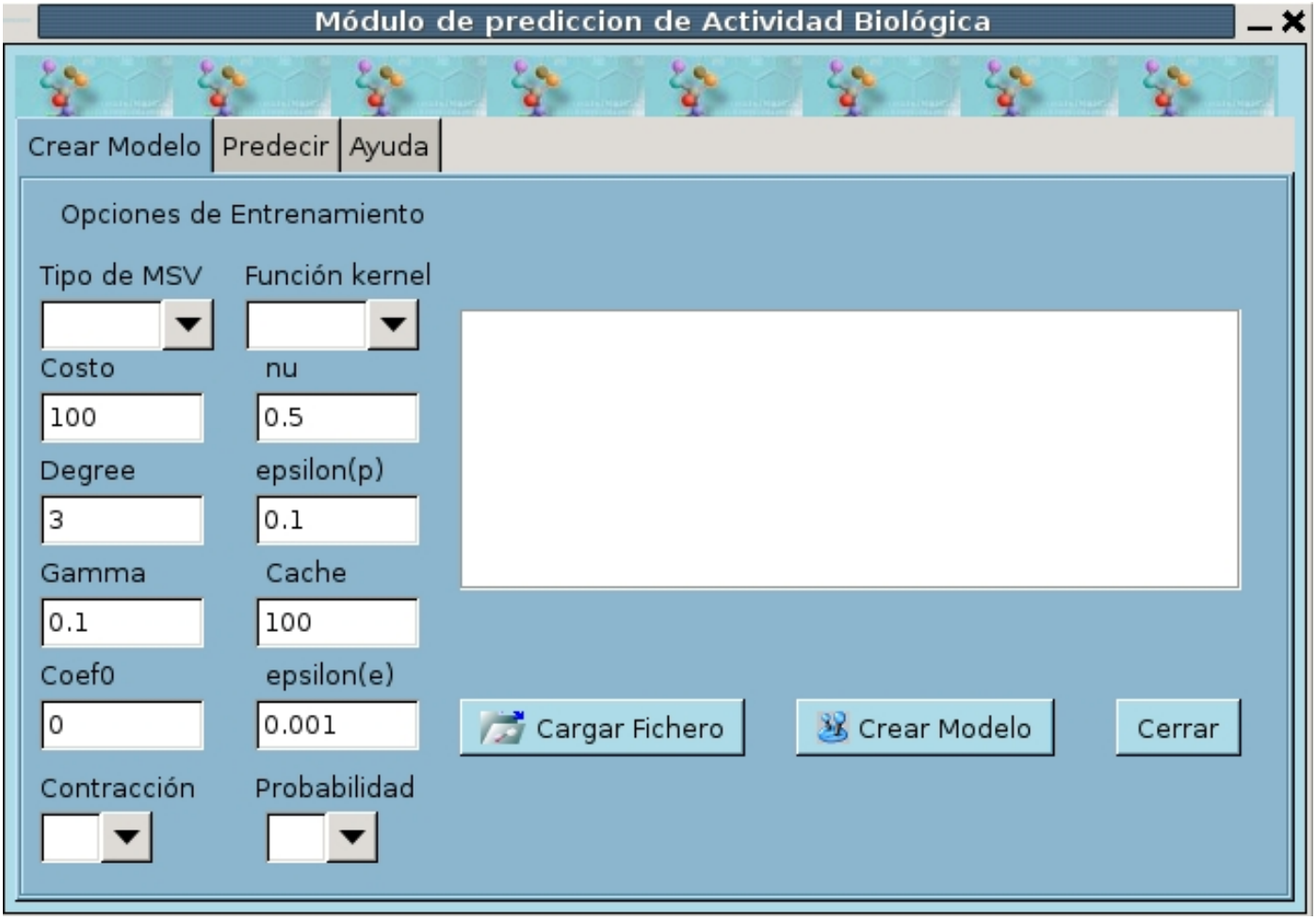



**Fig.: 4.2** Interfaz Crear Modelo.

Las pruebas realizadas en el presente trabajo de diploma se orientaron a validar la calidad del modelo y ver la capacidad de la MSV para clasificar los compuestos. En la tabla 4.1 se muestran los parámetros que se escogieron para cada prueba con sus resultados correspondientes. La MSV se crea con los valores escogidos y se entrena con el conjunto de entrenamiento. Solamente se modificaron los valores del costo de la función y el gamma del kernel. Una vez creado el modelo se predice utilizando la muestra de prueba.

Se realizaron varias pruebas, aunque sólo se muestran las de mejores resultados.

| Nro | Costo | Gamma | Mal Clasificadas |
|---|---|---|---|
| 1 | 100 | 0,1 | 28 |
| 2 | 1000 | 0,1 | 33 |
| 3 | 100 | 0,5 | 31 |
| 4 | 100 | 0,01 | 28 |
| 5 | 1000 | 0,5 | 26 |
| 6 | 10000 | 0,5 | 29 |
| 7 | 100000 | 0,5 | 29 |

**Tabla 4.1:** Valores de gamma y costo para la MSV c-svc y kernel FBR.

El modelo de clasificación que mejor predijo lo hizo con un 96.22% y 93.52% de acierto para las muestras de entrenamiento y de prueba respectivamente.

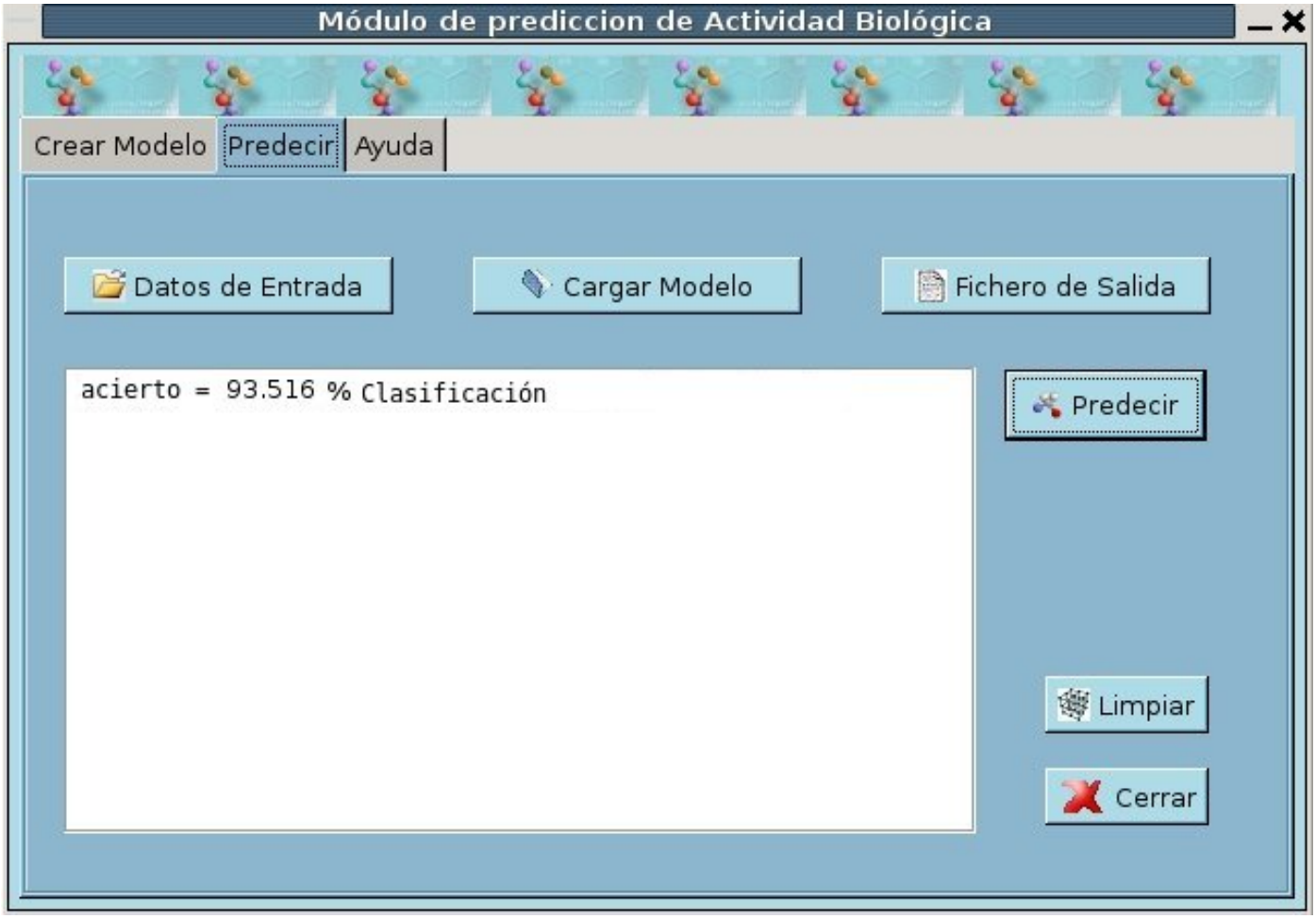



**Fig. 4.4**: Predicción para la muestra de Prueba.

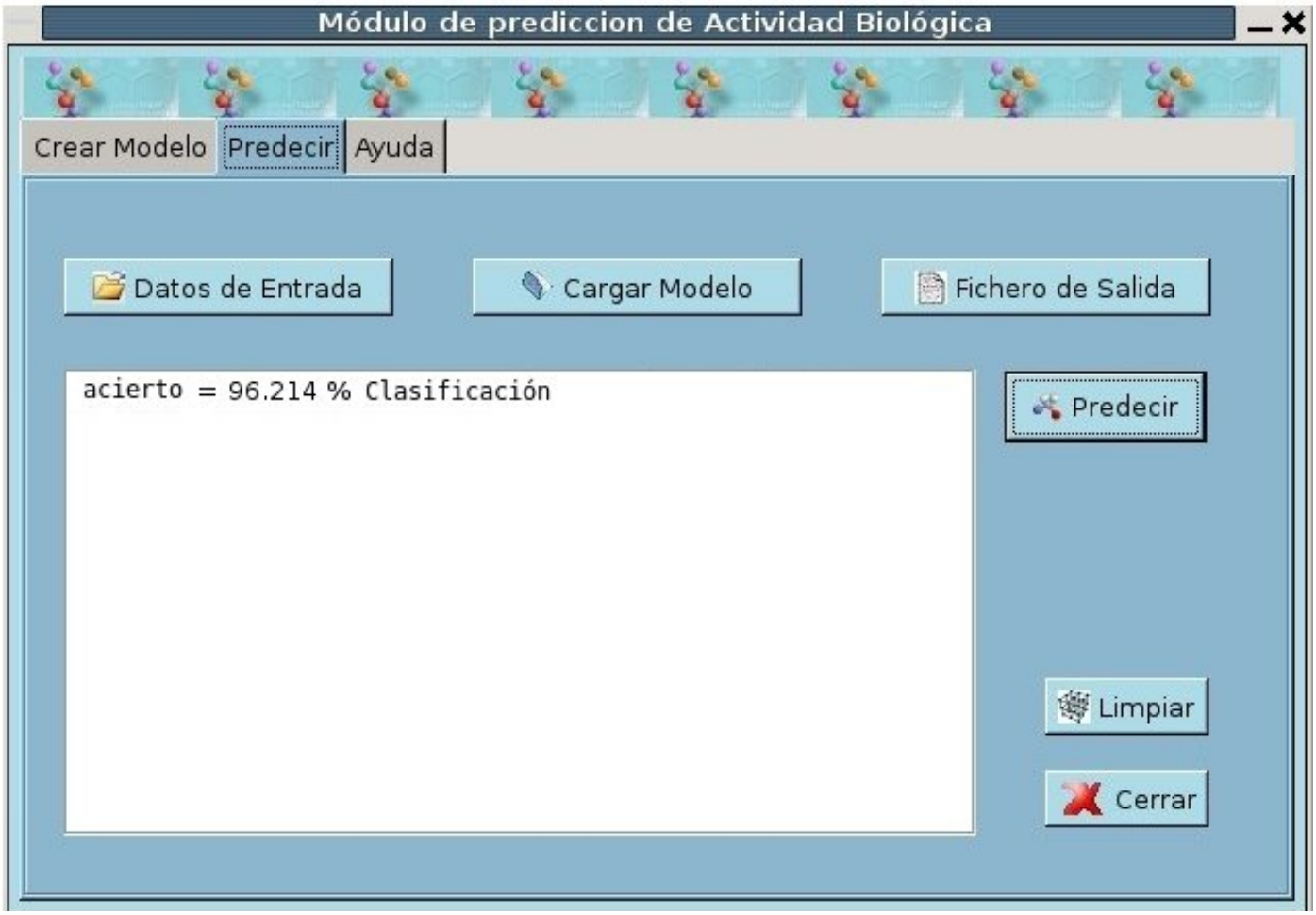



**Fig. 4.5:** Predicción para la muestra de Entrenamiento.

## Conclusiones

Se demostró la efectividad de las MSV para la creación de modelos de clasificación de moléculas orgánicas biológicamente activas.

Para este conjunto de datos de entrenamiento y teniendo en cuenta los resultados de las pruebas, los parámetros seleccionados, la MSV c-svc, el kernel FBR, Gamma = 0.5 y costo 1000, mostraron los mejores resultados

Se utilizó el diagrama de componentes para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre sus elementos.

## CONCLUSIONES

- Se analizó, diseño e implementó una aplicación informática para la predicción de actividad anticancerígena de compuestos orgánicos a partir de descriptores topológicos e híbridos utilizando Máquinas de Soporte Vectorial. Se corroboró que la combinación c-svc como MSV y de la FBR como función kernel se ajusta al problema planteado.

- Se elaboró un plug-in de la aplicación informática desarrollada para su incorporación al visualizador de la Plataforma.

- Con las pruebas realizadas a los modelos obtenidos por la aplicación en una clasificación de dos clases de compuestos anticancerígenos, se demostró la capacidad de predicción de las MSV siempre y cuando se escojan correctamente todos los parámetros necesarios para crearla.

## RECOMENDACIONES

Para aumentar las prestaciones que brinda la aplicación implementada se recomienda:

- Ampliar el tamaño de muestra e incrementar nuevos descriptores tanto topológicos y topográficos que permita generalizar los resultados.
- Utilizar el método de regresión.
- Implementar una funcionalidad que permita buscar los parámetros óptimos para el kernel y el costo de la función.

## BIBLIOGRAFÍA


1) *Manual de referencia de MySQL 5.0* 2007. [Disponible en: http://www.emagister.com/manual-referencia-mysql-5-0-cursos-2439962.htm
2) BELLERA, C. L.; A. TALEVI*, et al.* QUÍMICA COMPUTACIONAL APLICADA 2006.
3) CHANG, C.-C. and C.-J. LIN. *LIBSVM -- A Library for Support Vector Machines*, 2007. [Disponible en: http://www.csie.ntu.edu.tw/~cjlin/libsvm/
4) FARMACÉUTICA, C. D. Q. *SERVICIOS DE SINTESIS QUIMICA*, 2007. [Disponible en: http://www.cqf.sld.cu/esp/Servicios%20Cientifico-tecnicos/Servicios%20Cient-Tecnicos.htm
5) FOUNDATION, T. E. *Eclipse - an open development platform*. Disponible en: http://www.eclipse.org/
6) GARCÍA, V. E. S. *SELECCIÓN DE NUEVOS ANTIBACTERIANOS POR*
7) *TOPOLOGÍA MOLECULAR*. QUÍMICA FÍSICA, UV, 2003. p.
8) GONZÁLEZ, N. H. and I. T. BUSTAMANTE. *Las Máquinas de Vectores de Soporte*, 2007. p.
9) GUNN, S. R. *Support Vector Machines for Classification and Regression*, 1998. p.
10) HSU, C.-W. *LIBSVM Tools*, 2007. [Disponible en: http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/
11) HSU, C.-W.; C.-C. CHANG*, et al.* A Practical Guide to Support Vector Classification, 2007.
12) LU, N. C. W. and J. Y. G. LI. *Support Vector Machine in Chemistry*. Eorld Scientific, p. 981-238-922-9
13) LUCAS. *Curso de java* Disponible en: http://www.monografias.com/trabajos/java/java.shtml
14) MAK, G. THE IMPLEMENTATION OF SUPPORT VECTOR MACHINES USING THE SEQUENTIAL MINIMAL OPTIMIZATION ALGORITHM, 2000.
15) PORRAGAS, G. E. *Modelos QSPR/QSAR/QSTR basados en sistemas neuronales cognitivos*, 2004. [Disponible en: http://www.tdx.cesca.es/TDX-1016102-101509/
16) RAMÍREZ, M. L. D. C. G. *¿Qué sabes de los fármacos? Nuestras moléculas a prueba.* , Centro de Investigaciones Químicas, UAEM, 2006. [2007]. Disponible en: http://www.uaem.mx/posgrado/doctos/QSD_Farmacos.pdf

17) REMO. *Principales Ambientes de Desarrollo (IDEs)*, 2006. [Disponible en: http://www.comunidadjava.org/?q=node/44

18) RUBIO, J. L. D.; M. J. C. BLEDA*, et al. Discriminación y predicción de propiedades de fármacos mediante redes neuronales. Inteligencia artificial: Revista Iberoamericana de Inteligencia Artificial*, 2003. 7-16.

19) SANCHEZ, M. A. M. *Metodologías De Desarrollo De Software*, 2004. [Disponible en: http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

20) SCHOLKOPF, B. Statistical Learning and Kernel Methods, 2000.

21) SEMICHEN CODESSA, 1997.

22) SLÜTER, M. S. *APLICACIÓN DE DIFERENTES MODELOS DE REDES NEURONALES ARTIFICIALES (RNA), A LA SOLUCIÓN DE PROBLEMAS DE LA QUÍMICA ORGÁNICA, ESPECIALMENTE EN EL CAMPO DE LA QUÍMICA MÉDICA*. *REVISTA DE LA ASOCIACIÓN ESPAÑOLA DE CIENTÍFICOS* 2000.

23) SMOLA, A. J. *Support Vector Learning: Concepts and Algorithms*. Disponible en: http://sml.nicta.com.au/~smola/talks/iscas2001.pdf

24) SYSTEMS, P. S. A. *Modelado de Sistemas com UML*, 2000. [Disponible en: http://es.tldp.org/Tutoriales/doc-modelado-sistemas-UML/multiple-html/

25) TABERNERO, C. G.; R. B. PÉREZ*, et al. Construcción de un modelo de predicción para la propiedad farmacocinética excreción urinaria en fármacos orgánicos*. *MEDISAN*, 2001. 5: 16-23.

26) WELLING, M. Support Vector Regression.

27) YANG, H. *Margin Variations in Support Vector Regression for the Stock Market Prediction*. Department of Computer Science & Engineering, Chinese University of Hong Kong, 2003. p.

## GLOSARIO DE TÉRMINOS

**Especialista**: Cliente que utilizará el sistema.

**Trabajador**: Persona que se encarga de generar los nuevos modelos y realizar el entrenamiento de los mismos.

**Entrenamiento:** Acción que se realiza para el aprendizaje de la Máquina de Soporte Vectorial (MSV).

**Modelo**: Modelo de MSV que se crea a partir del proceso de entrenamiento para realizar la predicción.

**Predicción**: Acción que el sistema realiza para emitir un resultado.

**Descriptor**: Número que caracteriza estructuralmente la molécula.

**Actividad Biológica**: Es la actividad biológica de una molécula.

**Índice topológico**: Número que se calcula a partir de algoritmos que se aplican a la matriz de adyacencia o de conectividad del grafo molecular desprovisto de hidrógeno.

**Índice híbrido**: Número que se calcula a partir de algoritmos que se aplican a la matriz de adyacencia o de conectividad en el grafo completo donde los vértices de esa matriz de conectividad se ponderan con el valor de una propiedad químico-física del tipo de átomo que separan.

**Compuestos orgánicos:** Son sustancias químicas basadas en cadenas de carbono e hidrógeno. En muchos casos contienen oxígeno, y también nitrógeno, azufre, fósforo, boro y halógenos.

**Bioinformática:** Es la aplicación de los ordenadores y los métodos informáticos en el análisis de datos experimentales y simulación de los sistemas biológicos.

**CASE:** Computer Aided Software Engineering (Herramientas de ingeniería de software asistida por computadora).